UNIVERSIDAD NACIONAL DE LA PLATA

"ANÁLISIS DE PROXIES COOPERATIVOS EN SISTEMAS EDUCATIVOS"

(Observaciones y Mediciones realizadas en la Universidad de Karsruhe, República Federal de Alemania)

Eduardo Omar Sosa

Director: Lic. Francisco Javier Díaz

- 2002 -

A mi esposa e hijos.

AGRADECIMIENTOS

Al Director del presente trabajo, Lic. Javier Díaz de la Universidad de La Plata, a la Dra. Martina Zitterbart del Instituto de Telemática de la Universidad de Karlsruhe.

A mi esposa e hijos Angie y Milton por el apoyo durante el presente proyecto, fundamentalmente durante el período en el exterior.

A Julio, María Laura y Thyna por las reiteradas lecturas tras una correcta redacción y composición del mismo.

A la Universidad de Misiones (UNaM), la Universidad de Karlsruhe (UKA) y el Servicio Alemán de Intercambio Académico (DAAD) por la financiación y soporte de las tareas.

RESUMEN

La caracterización del tráfico en un servidor proxy en un servicio de redes es una ayuda en la determinación de parámetros de almacenamiento, la capacidad destinada a ese efecto y la necesidad de estudios de simulación. El caché a nivel local, o también llamado replicación, ayuda pero no alcanza en promedio al 16% del total de solicitudes, con lo que la mayoría de ellas debe ser completadas con solicitudes a los servidores originales.

En este trabajo se han realizado las observaciones y mediciones sobre cuatro servidores proxy cooperando entre ellos, que se encuentran actualmente en operación en la Universidad de Karlsruhe, República Federal de Alemania.

La información obtenida desde los archivos en los servidores instalados en los puntos de acceso de la red los cuales trabajan en forma transparente. Se ha hecho un estudio comparativo de los rendimientos de cooperación obtenida y los beneficios sobre la red y los usuarios. Si bien este servicio no es considerado primordial en la UKA, debido al ancho de banda que disponen, el estudio es pertinente para relacionarlos con implementaciones en nuestras Universidad Nacionales consideradas chicas, que deben considerar todos y cada uno de los métodos para optimizar el recurso más escaso al realizar el estudio, el ancho de banda.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

CAPÍTULO I: Introducción

1.1. - El problema

La otra cara de los beneficios de pertenecer a la red global, tiene que ver con el hecho pertenecer a una red insegura. Los servicios TCP/IP [**22**], [**23**], la variada complejidad de la configuración de los elementos de red, vulnerabilidades en el desarrollo de software, como así muchos otros factores contribuyen al riesgo de actividad de intrusos en las redes [**24**].

Previo a ahondar en el tema, es necesario que nos refiramos al punto central de todo análisis, que se convierte en la piedra angular de los temas relacionados a redes de datos: *la seguridad*.

Los términos *"seguridad de las redes"* y *"seguridad de la información"* se utilizan en el amplio sentido de la palabra respecto a que la información y los servicios prestados por una red, no puedan ser accedidos por un usuario no autorizado.

La provisión de la seguridad indicada anteriormente requiere de la protección tanto de los elementos físicos, como de los abstractos. Podemos destacar a los elementos de almacenamiento pasivo, como los discos rígidos, cintas de resguardo, ó discos compactos. También deben nombrarse a los elementos activos de una red, que son los usuarios finales de la misma.

En un ambiente de red, la seguridad física se extiende a los cables, bridges y enrutadores que comprenden la infraestructura de acuerdo a la topología de la misma. Obviamente, la seguridad física previene todo aquello relacionado con "escuchas indeseadas –wiretapping-".

La protección de un recurso abstracto como lo es la información, es mucho más difícil que la parte física, por la sencilla razón que este elemento es elusivo.

La *"integridad de los datos"*, como por ejemplo la protección de la información de cualquier cambio no autorizado, es crucial, como también lo es la *"disponibilidad de los datos"* a los usuarios legítimos evitando ser obviados por un ataque de denegación de servicios. Otros de los puntos importantes en la seguridad de redes es aquel conocido como *"garantía de privacidad"*.

Antes de tomar medidas referentes a la implementación de medidas de seguridad en las redes, debe determinarse que política se seguirá para la garantía de accesos a cada byte de información disponible en la misma, las reglas de diseminación de las mismas a otras personas y la forma en que la organización/empresa reaccionará ante un hecho de violación de la misma.

En Comer [**24**] se dice que *"los humanos son usualmente la parte más susceptible de fallar en cualquier esquema de seguridad, dado que un empleado malvado, resentido y sin cuidado de las normas de seguridad establecidas, comprometerá a la mejor seguridad implementada"*.

La única manera completamente segura de conectar servidores WWW a Internet, es aislarlos completamente de la red privada. Esto significa no permitir conexión alguna, física ni virtual, entre el servidor y la red considerada. Esto se transforma en un derroche innecesario de recursos, los cuales por lo generalmente son escasos y bastante onerosos. Para cumplir con éste propósito se utiliza una tecnología bastante difundida y accesible, que fuera denominada "Muro de Seguridad".

1.2. - Muros de Seguridad (Firewall)

Estas herramientas generalmente están compuestas por una colección de sistemas informáticos, routers, Computadores personales, o una combinación de cualquiera de estos elementos, en conjunto con la definición de políticas específicas.

Los muros de seguridad son simplemente una adaptación moderna del viejo sistema seguridad medieval: un foso profundo alrededor del castillo. Este diseño obligaba a cualquiera entrara o saliera del castillo a pasar por un solo puente levadizo, donde podía ser inspeccionado por la policía de Entrada/Salida. En las redes, es posible implementar la misma política. Una compañía ó empresa puede poseer muchas redes conectadas de manera arbitraria, pero todo el tráfico generado por ésta es obligado -direccionado- a pasar a través de un puente levadizo electrónico.

Etimológicamente, firewall proviene de la rama de la arquitectura, en donde se considera como muro de seguridad a aquel tapial lo suficientemente robusto y protegido de tal manera que permita a esa parte del edificio ser impenetrable a las llamas.

Un firewall, fuerza a todas las conexiones de la red a pasar a través de él, donde evalúa y examina el total del tráfico, proveyendo como primer medida a autenticaciones avanzadas que reemplazan al famoso esquema de "palabras claves", restringiendo el acceso a distintos sistemas, bloqueando servicios TCP/IP y otras medidas de seguridad adicionales.

En definitiva, la tarea de estos elementos es proveer el esquema seguro para las redes y subredes, permitiendo a los clientes y usuarios interactuar con Internet, solamente a través de él y en forma totalmente transparente; generando con ellos una política de *"Control de Acceso"* entre redes, por medio de dos mecanismos fundamentales: uno para bloquear el tráfico y otro para permitirlo.

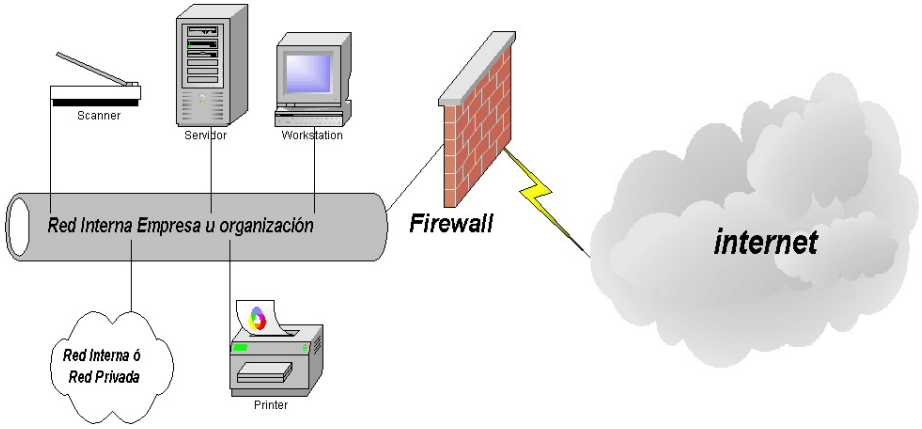

Figura 1.1 Esquema de una red configurada con firewall

Conceptualmente los dispositivos existentes se clasifican en aquellos funcionando a nivel de red y a nivel de aplicación. Los primeros ejecutan políticas basadas en direcciones IP fuente, direcciones IP destino y puertos involucrados. Ejemplo clásico es la aplicación de listas de acceso (filtros) y NAT (Network Address Translation) en routers. Estas herramientas generalmente son muy rápidas y transparentes para el usuario.

El segundo tipo de firewalls, son aplicaciones corriendo en hosts que no permiten el tráfico directo entre redes, realizando en primer lugar una auditoría y registro del tráfico en cuestión. Puede tener cierto impacto sobre la performance de la red, pero provee informes más detallados y permite la implementación de modelos más conservadores.

1.3. – Proxies

Los servidores proxy son hosts que ejecutan una aplicación para un protocolo, o conjunto de protocolos, controlando el tráfico entre una red privada e Internet.

El programa cliente del usuario interactúa con el servidor proxy en lugar de hacerlo con el servidor remoto instalado en Internet. El servidor proxy, evalúa la solicitud de cada cliente, y si esta es aprobada, inicia la transacción con el servidor remoto en nombre del cliente; de allí su nombre de proxy. Para el usuario este proceso es totalmente transparente. En estos hosts es donde se implementan en general los mecanismos de acceso al mundo exterior y por lo tanto los firewalls a nivel aplicativo.

Como conclusión podemos decir que un proxy WWW es una herramienta que permite el acceso al web de personas que se encuentran dentro de redes o subredes cerradas, implementando a su vez un firewall impidiendo el ingreso de intrusos.

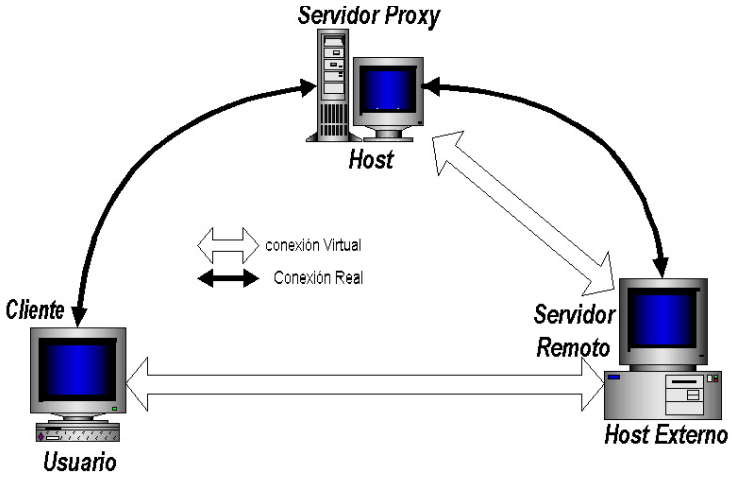

Figura 1.2 Conexiones Real y Virtual de un cliente interno a un host fuera de la red local

A la característica fundamental de los servidores proxy que es la de redirigir pedidos en nombre de diferentes clientes, se pueden considerar la implementación de servicios adicionales, que aumentan el rendimiento de la red, como lo es la capacidad de almacenamiento caché.

1.4. - El Caché

Tradicionalmente un caché es definido como un almacenamiento de objetos, de caracter temporario, de acceso rápido, a elementos usados frecuentemente.

Estos elementos pueden ser unidades de memoria de alta velocidad sobre microprocesadores, secciones de archivos en disco con unidades principales de memoria caché y caché en disco local.

El crecimiento de la red sobrecarga los servidores más conocidos, aumenta el tráfico en la red, y causa respuestas lentas a las solicitudes de los clientes. Una de las herramientas para menguar este inconveniente, es la utilización de proxies cachés.

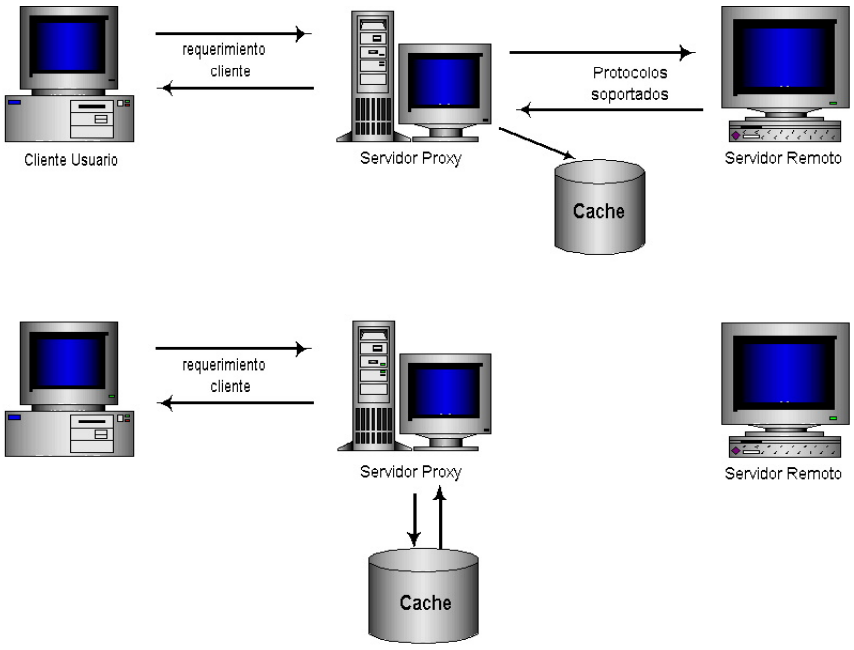

Figura 1.3 Mecanismo de funcionamiento de un servidor caché, en la resolución remota y local

El mecanismo de funcionamiento de dicha herramienta es exactamente el mismo que el efectuado en la utilización del caché de memoria. En el manejo de memoria, si lo solicitado está en una página aún no repuesta, es más simple utilizar esa copia existente, que tener que hacer una nueva solicitud, con el tiempo que conlleva el proceso, incluyendo las distintas fases como el cambio de contexto, paginado, etc.

Con los clientes web, ocurre exactamente lo mismo. Cuando un cliente solicita un documento, lo primero que realiza es una búsqueda en el caché propio lo que se conoce como localización temporal[1]; si no lo encuentra, obtiene el documento por primera vez solicitando esta información a un nivel más elevado de la jerarquía conocido como localización geográfica[2]. Una copia del objeto obtenido desde la página original, es almacenada en lo que llamamos el web caché. Las solicitudes posteriores del mismo documento son satisfechas con el envío de una copia del objeto directamente desde éste.

La búsqueda de métodos de mejoramiento de factores críticos para el servicio web, como la latencia, el tiempo de respuesta y conservación el ancho de banda cuando se accede a documentos en el WWW ha sido bastante intensa en los últimos años. Estas investigaciones han demostrado que la mayor parte del tiempo de respuesta de los requerimientos de servicios HTTP, se deben a problemas de red, más allá de aquellos que correspondieran al hardware impropiamente utilizado.

También es cierto que existen sitios en los cuales, la limitante no es precisamente la red, sino la capacidad de procesamiento. Este fenómeno se da fundamentalmente en lugares donde se produce un alto procesamiento de scripts dinámicos.

---

[1] Satisface los requerimientos de un mismo documento, por parte del mismo cliente.
[2] Servidores que satisfacen solicitudes de distintos clientes.

El fundamento de la implementación de estos servicios en la mayoría de la redes de la Internet, está basado en la premisa que es mucho más económico, pero principalmente mucho más rápido, obtener el objeto desde un lugar próximo al lugar de solicitud del objeto; que hacerlo desde algún sitio remoto de La Red. Lo básico y fundamental, es el almacenamiento de documentos populares[3] en servidores cercanos a los usuarios.

Muchos documentos son solicitados más de una vez, con lo cual el ancho de banda efectivo disminuye. ¿No es absurdo efectuar la transferencia del mismo archivo innumerable cantidad de veces, desperdiciando con ello en muchas ocasiones el bien mas preciado en nuestras redes, que es el ancho de banda?.

La lógica y racionalidad nos indica que, el archivo en cuestión debería ser transmitido solamente una vez a un servidor local, para después compartir esta copia con los usuarios restantes. Esta es la tarea que se encomienda a los servidores proxy.

1.5. - Limitante: ¿Red ó servidor?

Las dificultades en las redes, reconocidas por el usuario como retardos elevados en la obtención de los objetos web, poseen principalmente dos orígenes que son congestión en el tráfico ó valores de ancho de banda limitado en los enlaces entre el servidor y el cliente.

Los inconvenientes en los servidores se producen cuando estos están severamente cargados de solicitudes por lo que es sobrepasada la capacidad de procesamiento del procesador, o por rendimiento inadecuado del disco rígido.

---

[3] Documentos que son solicitados frecuentemente a lo largo del tiempo.

1.5.1 Los Protocolos y los usuarios

El tráfico generado por el protocolo HTTP es producido por la acción sobre hiperenlaces que son parte de las páginas HTML[4], como resultado de la acción generalmente se muestra otra página de éste tipo, imágenes; ó más aún cuando se acceden a páginas multimediales con contenidos de imagen y sonido.

Una páginas HTML contiene fundamentalmente, gráficos, fotos, texto formateado, audio y video. La diferencia fundamental con el tráfico generado por los otros servicios tradicionales en la Internet, es que estos como máximo pueden ofrecer y/o poseer es texto formateado.

La poca experiencia de la mayoría de los usuarios de los servicios disponibles en Internet, ergo para navegar por el WWW, ha diversificado la población de los usuarios; pudiéndose diferenciar entre los que utilizan éste exclusivamente, y los usuarios de otros tipos de servicios. El cambio en los hábitos de los usuarios, potencia aún mas la diversificación de las características propias de éstos y su comportamiento particular en la red.

Por lo general la solución de los administradores inexperimentados a los inconvenientes de tráfico encontrado en las redes, se responde con una de las siguientes acciones "pongamos un switch más grande, ampliemos nuestro ancho de banda, compremos más memoria RAM de la CPU ó precisamos inmediatamente un disco rígido más rápido y de mayor capacidad de almacenamiento".

1.6. - El Objeto Web

Un objeto Web consiste de un archivo HTML, conjuntamente con todos los archivos incluidos en él como referencias. La transferencia de todos estos archivos

---

[4] HTML Hypertext Markup Language

con los correspondientes tiempos de espera asociados se denomina "browsing session". El protocolo de transferencia de hipertextos posee elementos que lo relacionan con el Web Caché. Una solicitud HTTP esta formada por tres partes fundamentales: un método de solicitud, un URL (Uniform Resource Locator) y una secuencia de encabezados. Una respuesta HTTP consiste de un código numérico, una serie de encabezados, y el cuerpo de la respuesta.

El método más común de solicitud es el GET, que es una solicitud para obtener la información brindada por un dado URL. Aquí se observa la directa asociación del GET con la descarga. Otro de los comandos comunes del HTTP es el POST que se asocia con el colocar información en el servidor (upload).

Una de las tareas desarrolladas por los investigadores fue el desarrollo de la versión 1.1 del protocolo HTTP, bajo el supuesto que la red sea el cuello de botella en las transacciones en el ambiente web, persiguiendo reducir la latencia por medio de la disminución de los round-trips requeridos para transferir los objetos Web.

Sorprendentemente, se ha demostrado que dicha versión del protocolo se comporta mucho peor que su antecesor en condiciones de funcionamiento dadas.

1.7.- La cooperación de servidores

El caching, cuando es bien calculado y diseñado, ha demostrado ser una de las técnicas más eficientes para reducir la latencia, el congestionamiento de tráfico en redes, la carga en servidores y aumentar la confiabilidad de los sistemas.

Las arquitecturas actuales nos colocan en un dilema: compartir la información entre un gran número de clientes para obtener un porcentaje elevado de aciertos en el caché, acción con la cual estamos mejorando el rendimiento; mientras por otro lado, como ofrecemos el servicio a más clientes, estamos atentando contra el rendimiento del mismo dado que el tiempo de respuesta se incrementa proporcionalmente a la

distancia, a la mayor carga del caché y a el número de niveles en la jerarquía del caché.

Se han realizado exhaustivas investigaciones tendientes a demostrar el comportamiento de los cachés cooperativos, fundamentalmente con trabajos de simulaciones, pero muy pocas de ellas fueron realizadas en el inmenso escenario que corresponde al WWW.

En el presente trabajo, se pretende realizar una observación y comparación de dos sistemas educativos, conceptualmente idénticos; pero desde el punto de vista del tamaño, presupuesto, capacidad e infraestructura totalmente diferentes.

Mientras que el fin de la utilización de estas herramientas en la UKA persigue calidad y mejor servicio, en la UNaM se trata de optimizar el ancho de banda disponible para los usuarios, elemento sumamente escaso y muy valioso, máxime con relación con los flacos presupuestos universitarios y a los costos que estos servicios tienen en esta parte del mundo.

De la observación y la cuantificación de la utilización de cachés en la Universidad de Karlsruhe (UKA), República Federal de Alemania, deberíamos poder responder los siguientes interrogantes:

- Cuando es aplicada, ¿la cooperación se transforma en una mejora para el sistema? Si la respuesta es afirmativa, ¿ cuánto?

- ¿Cuándo debe ser utilizada la cooperación y cuándo no?

- ¿Cuáles son los efectos de la cooperación en los usuarios y la red?

Si bien las respuestas a estas preguntas serán apropiadas para la Universidad de referencia, de alguna manera habrá que establecer un paralelismo entre esta y la UNaM, contraparte de la Universidad europea en esta trabajo.

1.8 Financiación de las tareas

La visita de estudio al Instituto de Telemática de la UKA, se ha realizado por una invitación de autoridades de ese instituto y fue financiado en su totalidad por el Gobierno de la República Alemana a través del Servicio Alemán de Intercambio Académico (Deutscher Akadesmischer Austauschdienst DAAD).

CAPÍTULO II: Escenarios

2.1 La Universidad de Karlsruhe

En esta Universidad se encuentran instalados y funcionando 4 servidores Squid, configurados de tal manera que coopere cada uno de ellos con los otros tres.

Los servidores en la UKA, se encuentran instalados sobre una red montada sobre fibra óptica y ATM. La plataforma utilizada es PC+LINUX en todos los casos, utilizándose la distribución Linux realizada por Debian. Como servidor Caché se instalaron paquetes Squid versión 2.4. Los equipos en todos los casos eran integrados por un procesador Pentium III, y una capacidad de 20 GB[5] en disco Rígido. El ancho de banda en el campus es de 622 Mbps[6].

La conectividad de la UKA está asegurada por 4 enlaces de 2,4 Gbps[7] cada uno de ellos, uno directo a Internet; y los otros tres contra los nodos de la red Universitaria de datos, en las universidades de Stuttgart, Heidelberg y Freiburg. En cada uno de los puntos de acceso a la red de la UKA, se ha instalado un servidor Squid como servidor proxy. Las Facultades e Institutos aceden a esta infraestructura por interconexión de redes fast-ethernet a 100 Mbps. Esta tecnología será actualizada a Giga-ethernet durante el corriente año. Inclusive se encuentran instalados vínculos con las residencias estudiantiles, por ejemplo a HADIKO[8].

La cantidad de direcciones IP detectadas en los archivos de registro alcanzan las 1.500, por lo que se estima que atiende a una población que ronda los 16.000 (dieciséis mil) usuarios.

---

[5] GB: Gigabyte, constituído por 1.024 Megagytes (MB).
[6] Mbps: Megabits por segundo.
[7] Gbps: Gigabits por segundo
[8] HADIKO: Hans Dieckman Kolleg, residencia estudiantil de administración descentralizada de la Universidad de Karlsruhe. http://www.hadiko.de

2.2 La Universidad de Misiones

La Red de Datos de la Universidad Nacional de Misiones se encuentra operativa desde el año 1986. La misma nació como una inquietud de las autoridades de esa época en la UNaM por contar con el servicio de correo electrónico. Se implementó un servicio UUCP con la UBA[9], quien se presentaba a sí misma como MX[10] primario.

Todas las dependencias de la UNaM se encuentran interconectadas por medios sobre distintas tecnologías para llegar a cada uno de estos puntos en la Provincia de Misiones. Para nombrar los medios utilizados con este fin podemos decir que las sedes de la ciudad de Apóstoles y Oberá se encuentran vinculas con la red digital soporte, bajo una trama Frame-Relay sobre la cual se habilitó asimismo VOFR[11]. Dado que la ciudad de Eldorado no se encuentra vinculada por la RDS[12], se ha hecho necesario instalar un vínculo satelital. El ancho de banda disponible a las sedes de Eldorado y Oberá es de 256 Kbps[13]. A la ciudad de Apóstoles de 128 Kbps.

Estas capacidades de transporte de datos están referidas a su interconexión con el centro de la estrella ubicado en el Campus Universitario sito en la afueras de la ciudad de Posadas. Las Facultades e Institutos del centro de la ciudad de Posadas se interconectan con el Campus, por medio de enlaces propietarios sobre microondas a 2 Mbps.

La salida nacional/internacional se produce por el vínculo operativo de la Red de Interconexión Universitaria (RIU) desde el Campus Universitario a Telecom Buenos Aires, con un ancho de banda contratado de 256 Kbps.

Este ancho de anda disponible, es motivo de reclamos y quejas continuas de los usuarios de la Red de Datos de la UNaM por los retardos y bajos rendimientos de las redes interconectadas.

---

[9] Universidad de Buenos Aires
[10] Mail Exchanger, equipo receptor de mensajes para un determinado dominio.
[11] Voice over Frame Relay, telefonía interna de la UNaM sobre la red de datos.
[12] Red Digital Soporte.
[13] Kilobits por segundo

En rectorado se ha instalado un servidor Squid, sobre plataforma i386+Linux con 8 Gbytes de capacidad en disco, 512 MB de RAM y procesador PIII de 800 MHz como una medida mitigatoria de los efectos de la falta de capacidad de transporte. La utilización del mismo por los usuarios de la UNaM no es obligatoria; esto se debe a la política de descentralización de la administración de la Red implementada oportunamente, esto equivale a decir que los clientes del caché pueden estar accediendo al mismo por satélite, radio ó cable simultáneamente.

Atento a la necesidad de incrementar su capacidad de transporte, a mediados del mes de Agosto de 2002 se instaló en la Facultad de Ciencias Exactas un vínculo a la red pública de banda ancha (ADSL) de 512 Kbps. Sobre este NAP[14] se ha configurado un servidor idéntico al del Campus, con servidor caché Squid e IPTables. Este servidor caché se configuró de manera de trabajar en forma cooperativa con el instalado en Rectorado. Los resultados de esta configuración podrán evaluarse una vez que el caché haya entrado en régimen, con los archivos de registro depurados.

[14] Network Access Point, Punto de acceso a la red

CAPÍTULO III: Antecedentes

3.1- Antecedentes Encontrados

Muchos sistemas en Internet utilizan directorios de metadatos o multicast para localizar la información solicitada para luego realizar una transferencia caché-caché. La diferencia fundamental entre todos ellos es la forma en la cual se estructuran éstos directorios.

Durante los últimos años se han realizado exhaustivas investigaciones sobre el modo de lograr la cooperación de proxies a fin de poder aumentar la población efectiva de clientes, incrementar el índice de aciertos (HIT ratio) y fundamentalmente reducir la latencia de acceso a distintos documentos.

Las arquitecturas caché se diferencian entre jerárquicas y distribuidas. En la primera de ellas los servidores son localizados en diferentes niveles a través de la red. En la arquitectura distribuida, los servidores son instalados en el último nivel de la red y no existen servidores intermedios.

La cooperación entre proxies, fue propuesta inicialmente en el proyecto HARVEST que desarrolló el protocolo ICP (Internet Cache Protocol) utilizado en la búsqueda y descarga de documentos desde caché vecinos [**3,4**]. Una variante popular de este proyecto que se puede obtener de Internet es el paquete Squid [**5**]; que ha sido utilizado en el presente trabajo.

En los Estados Unidos de América, The National Laboratory for Applied Network Research (NLANR) [**11**] posee un caché jerárquico global consistente en 8 proxys padres, que atienden a cientos de proxies cada uno. La organización de estos sistemas es estática y en general estos son configurados manualmente.

Las estructuras de los cachés distribuidos se dividen en dos grandes grupos:

- ✓ los que utilizan un servidor estático para decidir cuando, como y donde cachear, donde los clientes solicitan directamente al servidor web la información, y este decide cual es el servidor caché más cercano al cliente por comparación geográfica o por topología de red (conteo de saltos)

- ✓ aquellos que utilizan el caché por diseminación y mantienen los metadatos en directorios que deben ser descubiertos por los usuarios.

Los primeros, primeros fueron tratados por Gwertzman y Seltzer [**12**], mientras que los del último tipo fueron investigados por Bestavros y Cunha [**23**].

En trabajos posteriores fue demostrado que el sistema de hop counting y de distancia geográfica definido por Gwertzman y Seltzer no podía predecir la latencia en el tráfico de Internet. El push caching tuvo numerosos problemas de implementación tanto políticos como así también técnicos, debido a lo cual tuvo poco éxito. A ello debe adicionarse que los beneficios son insignificantes respecto a servidores caché individuales.

En el caso de directorios de metadatos, la información de cada objeto transferido es almacenada en cada proxy y puede ser propagado separadamente de los datos del mismo objeto. Si bien el despacho de los objetos en una red, utiliza una transferencia simple entre el servidor y el cliente; los metadatos pueden ser propagado de diferentes maneras. Algunos propusieron una estructura de "push" jerárquico, mientras que otra doctrina sugería la compresión de los metadatos en un "caché digest" para ser intercambiado con otros proxies [**8**]. El CRISP (Caching and Replication for Internet Service Performance) [**10**] utiliza un directorio global centralizado. Actualmente existen numerosos cachés jerárquicos cooperando bajo el protocolo ICP en el laboratorio National Lab of Applied Network Research [**5**].

Otros proyectos pretenden que los cachés cooperen bajo condiciones diferentes, como el CARP (Caché Array Routing Protocol) [**6**] que describe un protocolo distribuido de Caching, basado en una lista conocida de proxies y una función de tachado para dividir la distancia entre estos servidores. Como se observa, para a ser una alternativa para ICP, donde todas las consultas se realizan por HTTP; no utilizando otra capa de aplicación de protocolo como lo realizado por ICP, situación sobre la cual puede utilizarse con mayores ventajas los encabezados de HTTP versión 1.1.

Summary Caché [**7**], es un proyecto donde cada proxy mantiene un resumen de cada uno de los directorios caché de los proxies participantes. Se busca en estos resúmenes, antes de hacer ningún requerimiento. Los autores argumentan que existen

2 factores para que este protocolo posea un bajo overhead: que los resúmenes son actualizados periódicamente, y que la representación de los directorios es simple, tan bajo como ocho bits por entrada.

En el Proyecto Relais [**9**] el protocolo permite cooperar a grupos de cachés distribuidos geográficamente garantizando a sus clientes una actualización monotónica y rápida de las versiones de los documentos.

Todos estos proyectos usan diferentes estrategias para compartir meta información indicando la ubicación de los documentos buscados en un Proxy.

Una tercera línea de trabajo los propuso como grupos multicast, tal es el caso del AWC (Adaptative Web Caching) que fuera presentado por Zhang et al. [**13**]. Estos autores sugieren la utilización de multicast para configurar automáticamente los grupos caché. Los requerimientos de datos son enviados a una dirección multicast, con lo cual los clientes no tienen que reconfigurar su sistema cada vez que un miembro ingresa o egresa del grupo. Los proxies en este caso son particionados en grupos y los objetos son obtenidos desde el servidor original a través de un camino de grupos solapados. Consecuentemente, AWC requiere mucha transferencia de información entre redes remotas, exactamente igual que con los cachés jerárquicos.

Otras publicaciones sobre la performance de los proxies cooperativos, fueron enfocadas hacia la forma de cooperar y el rango de HITS. Alguno de ellos han demostrado que la relación de HITS para proxies caché crece en forma logarítmica con el número de clientes y de solicitudes. Para el caso de servidores atendiendo una poca cantidad de usuarios, la relación de HITS aumenta rápidamente a medida que lo hace el número de usuarios. Sin embargo la mejora va disminuyendo, y luego de cierto punto, la relación de HITS se vuelve independiente del aumento de la población.

Algunos han aseverado que la cooperación entre cachés aumenta la relación de HITS, para poblaciones menores a los 20.000 usuarios, pero ofrece muy poca mejora desde este límite. Estos resultados a su vez sugerían que basados solamente en la cantidad de HITS, la cooperación a gran escala solo tiene sentido en ambientes con amplios anchos de banda y baja latencia.

La mayoría de los estudios de proxies caché se basan en el uso del HIT como unidad de respuesta, mas allá de ser establecido el tiempo de respuesta o aumento en la velocidad; por lo que en todos ellos no queda finalmente muy claro si por efecto de la cooperación se puede mejorar el tiempo de respuesta.

CAPÍTULO IV: HTTP y caching

4.1 Características de HTTP

El protocolo de hipertexto (HTTP) posee características particulares que permiten la implementación del caching. Como se sabe, una solicitud HTTP está compuesta por tres partes fundamentales: el método del requerimiento, el Uniform Resource Locator (URL) y un grupo de encabezados. Una respuesta HTTP consiste en un código numérico, grupo de encabezados y cuerpo de repuesta opcional. Los métodos tradicionales de transacciones son el GET y el POST, los cuales están relacionados con la obtención y el envío de datos.

Una forma especial de GET es el condicional, que es diferenciado por la inclusión de un encabezado *If-Modified-Since*, Esta es la herramienta fundamental del protocolo HTTP que permite la implementación del caching, con el cual se permite a los servidores enviar una respuesta No Modificado, a partir de lo cual el cliente utilizará la copia almacenada en su caché de disco. Si así no lo fuere, el servidor envía una nueva copia del documento.

El HTTP v1.1 posee el encabezado *Caché-control* que permite a los clientes y servidores, obviar los algoritmos normales de cacheo. En este protocolo existe un directiva *Max-age* con la cual los clientes establecen el tiempo máximo que almacenará el objeto en el caché propio, sin efectuar una recarga del mismo. Excedido este, si la copia almacenada en el servidor es más antigua que la solicitada por el cliente, el mismo es obtenido directamente de los servidores originales.

4.2 Caching

Desde la masificación del uso del World Wide Web, alrededor de 1994, mucho trabajo y esfuerzo se ha dedicado a la lucha contra la latencia experimentada por los usuarios. Este fenómeno experimentado en los diferentes sistemas puede producirse por diferentes razones, tales como problemas de sobrecarga de enlaces, limitaciones de ancho de banda, etc.. Too esto nos hace recordar conceptos básico de redes, tal como aquel que dice que *"una transacción es lo suficientemente buena como lo es su elemento de comunicación más débil"* [**15**].

El caching, a nivel local, demuestra su utilidad cuando un documento es accedido por el mismo usuario varias veces. Allí, el sistema del usuario, utilizando un caché local en la máquina, guarda una copia de los documentos obtenidos, por un determinado período de tiempo, pudiendo esté obtener una copia del mismo directamente de la estación local, sin originar tráfico de red alguno.

4.3 Proxies e ICP

El comportamiento anterior, ha sido copiado por los creadores del ICP para permitir a un grupo de usuarios, poder acceder a documentos obtenidos anteriormente por algún otro vecino de la red. Es lo que llamamos un caché de red compartido.

En el World Wide Web se conocen a estas herramientas como proxies. Los proxies son utilizados también generalmente como pasarelas (gateways) entre los dos lados de un firewall; las redes privadas y públicas, no necesariamente realizando caching (ver Capítulo I).

Usualmente el proxy es utilizado por todos los clientes en una subred. Esto permite el caching de información solicitada por cada uno de los usuarios/clientes de la misma.

El problema que presenta el cacheo, es que el contenido almacenado puede ser lo suficientemente antiguo como para no corresponderse con la información actual del servidor de referencia. Dado que las modificaciones en el contenido de los sitios son impredecibles, es casi imposible determinar un tiempo de vida para la información en el caché. El no utilizar proxies, no es una buena idea.

Lo fundamental en la instalación de servidores caché, es que estos deben estar insertos en el camino hacia el servidor original. Hay dos formas fundamentales de lograr estos efectos. Una de ellas, es la configuración del software de cada uno de los clientes para que las solicitudes sean enviadas al servidor proxy. Esta función es soportada por todos los navegadores (Internet Explorer, Netscape, Mozilla, Opera,

etc). Otra forma es utilizar equipos de capa 4 de OSI, para que realice un proxy transparente. En este caso todas las solicitudes son enviadas por el equipo al servidor proxy. Si el proxy no es capaz de resolver la solicitud, este realiza el pedido directo al servidor de origen.

4.4 ICP: Características y Definiciones

Definimos a ICP como un formato de mensaje utilizado para la comunicación entre cachés de objetos web. Estos mensajes son enviados a servidores vecinos, preguntando por un objeto determinado. Estos vecinos responden simplemente indicando *HIT* si lo posee o negándolo con *MISS.*

En la práctica, ICP está montado sobre UDP, pero no es una condición necesaria. Las razones del uso de este servicio se basan en que una mensaje ICP de requerimiento/respuesta necesita completarse rápidamente, por defecto dentro de los dos segundos de emitido. El caché no puede esperar mas que ese tiempo para empezar a descargar el objeto solicitado. Una falta de respuesta a esta solicitud generalmente significa que la ruta esta congestionada o el enlace caído. Esta característica de UDP, a diferencia de TCP, permite conocer las condiciones de la red, haciendo de este método el más apropiado para la implementación de ICP.

Aparte de su utilización como protocolo de localización de objetos almacenados, los mensajes ICP pueden utilizarse para la selección de estos servidores caché. Una falta de respuesta a un requerimiento puede indicar una falla en el sistema o en la red. La respuesta ICP puede incluir información, a priori, sobre que servidores utilizar para la obtención de los objetos en cuestión.

Una de las marcadas desventajas de este protocolo, según diferentes autores, es el overhead del mensaje. Por cada MISS en un caché determinado, debe enviarse un mensaje ICP a todos los servidores hermanos consultando sobre el mismo objeto. Cada sibling (hermano), debe procesarlo y contestarlo. Aparte de esto, los objetos

más populares; es decir los de acceso más frecuente, se encontrarán replicados en varios servidores conllevando a un uso no racional de memoria y capacidad de discos rígidos.

El tráfico extra creado por un proxy se puede calcular como el número de MISS en modo standalone, multiplicado por el número de proxies con quien coopera. El tráfico extra en la red, es la suma del tráfico por cada proxy y el tráfico extra de cada proxy. Se puede definir así al porcentaje útil de tráfico generado por cada proxy como la relación del número de solicitudes generadas por el mismo que son servidas por otros proxies al tráfico extra creado por este. Entonces podemos decir que, definimos al tráfico útil para la red, como la relación del incremento en el número de HITS sobre todos los proxies, al tráfico generado sobre la red.

4.5 Relación entre ICP y HTTP

Debe dejarse bien en claro que el ICP es un protocolo extremadamente liviano. Considérese que los encabezados están en formato binario para que sea mas compacto y el payload del mensaje generalmente es un URL. Como es de esperarse, esto tiene un lado positivo y uno negativo.

Un caché recibiendo un ICP_QUERY tiene solamente que extraer el URL, buscar la existencia de ese objeto en sus directorios y elaborar la respuesta. Téngase en cuenta que del roundtrip del ICP deberá ser bastante rápido, dado que el caché debe manejar muchos más pedidos ICP que HTTP. Según publicaciones del NLANR, esta relación se encuentra afectada por un factor de 4 a 6. Esta característica debe cumplirse, para evitar que las solicitudes ICP sobrecarguen el proceso de caching, y para minimizar los retardos dentro de la estructura jerárquica.

La desventaja principal del ICP, es que no puede manejar HTTP. Los cachés utilizan ICP para localizar los objetos, pero después deben utilizar HTTP para obtenerlos.

Estas diferencias entre ICP y HTTP han sido discutidas, y aún continúan siéndolo. La discusión principal en estos foros, trata sobre si el protocolo ICP debe seguir manteniéndose con un protocolo liviano ó es ya momento de efectuar sobre el mismo modificaciones tendiente a obtener una nueva versión con funcionalidades de HTTP.

Con estas características, una consulta ICP permitiría no solamente la ubicación del objeto deseado, sino que se podría especificar la solicitud HTTP completa mas allá que el URL. Sin embargo, dicha modificación precisaría de más ciclos de CPU para la realización de la tarea.

Si se siguiera con esta tendencia, se podría llegar a modificar al ICP, para llegar a conocerlo como HTTP sobre UDP, para lo cual habría que realizar una definición de un nuevo método de solicitud (query) y unos pocos códigos de status.

4.6 Definiciones básicas de Web Caching

Existen en la literatura, numerosas referencias que suelen ser interpretadas de distintas maneras. Por ello se considera conveniente realizar un compendio de la terminología.

Los términos incluidos en esta tabla corresponden a aquellos más frecuentemente utilizados en la bibliografía.

Tabla 4.1 Términos utilizados normalmente al considerar el cacheo de objetos WWW

Término	Definición
Usuario	Proceso de paginado que envía la solicitud original utilizando el protocolo HTTP
Servidor Proxy	Gateway a Internet y Caché HTTP para un grupo dado de usuarios. Un servidor proxy, comparte los objetos cacheados con los miembros pertenecientes al grupo. Las solicitudes de estos son enviadas al servidor Caché. El proxy server usualmente se encuentra en la misma red que los usuarios.
Cliente	Uno de los usuarios del servidor proxy
Servidor	Servidor HTTP al cual pertenece el objeto solicitado
Caché Usuario	Objeto caché HTTP para un usuario determinado, el cual está localizado en el misma máquina que él
Caché cliente	Puede ser un caché usuario o proxy caché
Caché remoto	Caché que no se encuentra en la misma red que los clientes.

Estas definiciones son clarificadas en el siguiente gráfico, teniendo en cuenta que el caching está montado sobre una arquitectura cliente-servidor, pudiendo estar los cachés, en el servidor, el cliente o sitios intermedios.

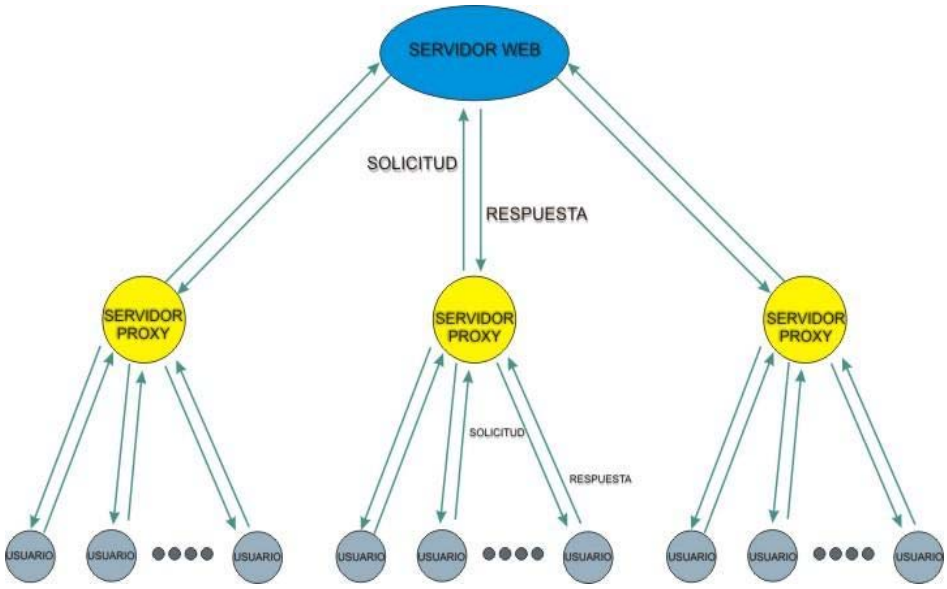

Figura 4.1 Rol de del servidor caché en una estructura jerárquica

En el gráfico puede apreciarse el servicio prestado por los servidores proxy, como intermediarios entre las solicitudes de los clientes y servidores en sus locaciones físicas respectivas. La solicitud de un cliente podrá ser obtenida desde el servidor proxy de la intranet ó desde su servidor primario.

4.7 Caching Jerárquico e ICP

4.7.1 Parents, sibling e ICP

La aplicación de un método jerárquico no es algo nuevo en los servicios de Internet utilizados en áreas extensas. Basta considerar que el mismo esquema es aplicado por el Sistema de Nombres (DNS), newsgroup Usenet y CIDR (Classless Inter-Domain Routing).

El uso de esta estructura es muy beneficioso por la escalabilidad experimentada y el manejo del caching que es posible realizar.

En un sistema de caché jerárquico simple imaginemos a las tres Facultades de Posadas (Humanidades, Químicas y Económicas) habiendo instalado un caché en cada uno de sus sitios, han configurado a Rectorado como caché padre. Las solicitudes que no son satisfechas por los cachés locales, son remitidas al servidor de Rectorado, si este no posee la información requerida es solicitada directamente al servidor de origen.

Dicha topología no siempre es válida para satisfacer todas las necesidades. Por ejemplo, podrían existir muchos cachés padres. En ese caso, ¿a cual de ellos se debería remitir la solicitud y cual debería ser la información disponible para ayudar al caché en esta decisión? O, por otro lado, podría darse el caso que no existiera caché padre alguno.

Consideremos los tres cachés anteriores, y que, por alguna razón las Facultades no desean ser dependientes del caché de rectorado. Las facultades querrán

consultar a los cachés de sus pares. Pero: ¿Cómo saben que documentos están disponibles para consulta y que lenguaje debe utilizarse para la consulta?

El ICP viene a cumplir ese cometido, rápida y eficientemente, ofreciendo mecanismos para el establecimiento de jerarquías de caché complejas. ICP permite consultas entre los cachés hijos, llamándolas a estas *relaciones sibling*. La diferencia sustancial entre parent y sibling (padres y hermanos) se establece al momento de no poseer una respuesta al requerimiento de un cliente. Un padre, puede ayudar a resolverlo. Un hermano no. La jerarquía en un sistema caché se esclarece en la Figura 4.2.

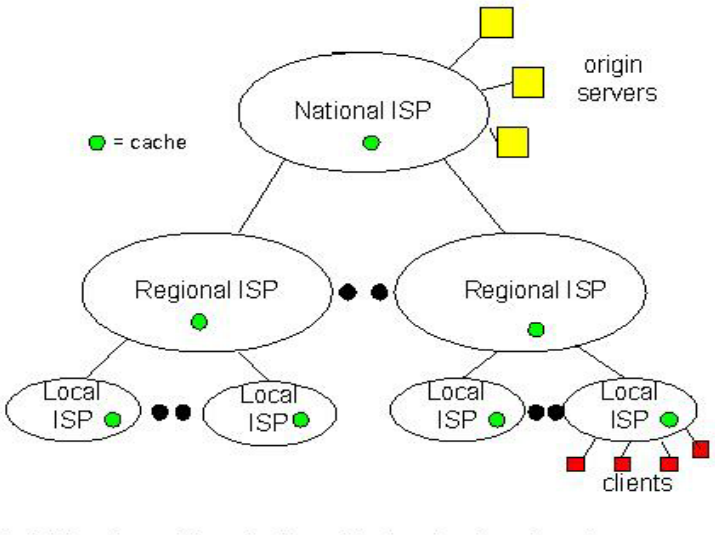

Figura 4.2 Esquema nacional de caché

4.8 ICP y SQUID

Al momento de proponerse las arquitecturas a utilizarse en el caching, muchas fueron las que se presentaron como posibles alternativas. Una de ellas fue el Squid Web Caché System, sistema que a partir de entonces se ha convertido en el standard "de facto" para el Web Caching.

Si bien el paquete Squid ha alcanzado sus objetivos con creces, y que sigue evolucionando, existen autores e investigadores que piensan en un sistema distinto y adaptable a la diferentes necesidades y situaciones, los cuales proveerían una salida

efectiva y natural hacia sistemas de caché por demanda. Estos deberán necesariamente ser robustos, eficientes y escalables.

La infraestructura dominante en la Internet de hoy día, consiste en una estructura jerárquica de Objetos Caché Squid.

En estos sistemas, todos los cachés pueden estar interconectados y deben ser configurados manualmente en un árbol jerárquico. Cada servidor caché intercambia mensaje ICP con sus peers (parents y siblings) para determinar cual de ellos es el más apropiado para la provisión del objeto buscado, ya sea a través de una copia en su caché propio o obtenerlo directamente desde su servidor original.

Fundamentado en el suceso de Squid, numerosas investigaciones apuntaron a las mejoras que deberían realizarse en el paquete para que el mismo pueda considerarse como de aceptación total.

Lo primero que se determinó como primordial, es eliminar el tedioso trabajo manual de los administradores, por razones configuración, los cuales deben hacerlo mancomunadamente, para que la jerarquía funcione adecuadamente. La segunda y aún más importante, es lograr que el sistema sea menos sensible a los errores humanos y de administración del sistema en aquello que tiene que ver con el comportamiento general del mismo. Según diferentes autores, la exclusiva configuración manual de estos paquetes, menoscaba todo lo atinente a escalabilidad y cercena la habilidad de los mismos a adaptarse a las condiciones cambiantes de las redes. Otra de las mejoras que se ha planteado sobre el paquete, es evitar el delay producido por el protocolo de saludo de tres vías utilizado para el intercambio de mensajes ICP.

Un requerimiento atendido por un caché lo denominaremos HIT mientras uno que no lo es lo llamaremos MISS. Estos últimos pueden poseer las siguientes características:

•**Primer Acceso:** El documento es obtenido por primera vez.

- **Capacidad:** El MISS ocurre cuando se está accediendo al documento en el caché, pero que es descartado por limitaciones de espacio

- **Actualizaciones:** Se solicita un documento previamente obtenido, pero el mismo ha caducado durante el intervalo

- **No almacenables:** Cuando se pretenden documentos que no son cacheables, tales como documentos dinámicos generados por scripts cgi-bin, o documentos que cambian su contenido muy rápidamente.

Aún cuando el Caché a utilizar tenga una capacidad de almacenamiento infinita, el número de MISS en un caché institucional puede llegar a ser tan alto como el 50 al 70 % [**1,2**].

Otros estudios más recientes han demostrado que los documentos no almacenables se encuentran entre el 10 y el 20% de total de los documentos solicitados. Esto se cumple siempre que la población de clientes que atiende el caché sea pequeña.

4.8.1 La cooperación entre cachés

Las características que determinan la utilización de este tipo de herramientas, se fundamentan en que los usuarios tienden a acceder un gran número de sitios, en los cuales permanece un corto período de tiempo; por lo tanto el porcentaje de uso del caché por usuario es bajo. Por esta razón muchas organizaciones han comenzado a utilizar proxies cooperativos, con lo cual los usuarios obtienen el beneficio de utilizar objetos anteriormente procesados por otros.

Como conclusiones de otros trabajos sobre la cooperación entre proxies se puede nombrar los dos puntos siguientes:

1. *El caché cooperativo es efectivo solamente cuando la densidad media de acceso es relativamente alta. Definimos este parámetro como la relación entre el número de requerimientos al número de objetos distintos en una ventana de tiempo.*

2. *La efectividad de este sistema decrece a medida que el sesgo el la popularidad de objetos aumenta. Valores elevados de este término (sesgo), significan que solamente un reducido número de objetos son accedidos más frecuentemente, reduciéndose el beneficio obtenido de grandes cachés, y con ello de la cooperación.*

¿Pero cuál es el mecanismo utilizado por el software Squid para realizar una consulta ó responder a una de estas?

4.8.2 Algoritmo de Solicitud ICP

Bajo un comportamiento standard, Squid envía un mensaje **ICP_QUERY** a cada uno de los peers existente. Esto puede conducir a situaciones indeseables. Por ello existe la posibilidad de configurarlo de distintas maneras. La más frecuente es aquella en la que Squid envía su solicitud ICP a hosts específicos dentro de un dominio determinado en el DNS.

Otro parámetro de configuración de Squid *hierarchy_stoplist*, el cual permite excluir cierto tipos de solicitudes. Comúnmente, se obvian las solicitudes a URLs conteniendo la cadena *"cgi-bin"* o el signo de pregunta(?), dado que estos indican que son no cacheables, o son páginas dinámicas con lo que la probabilidad de un ICP_HIT es muy baja.

Para cada solicitud que no es filtrada por el hierarchy_stoplist, el programa envía una solicitud a cada peer, a menos que:

- *una regla en **caché_host_domain** no permita el uso de ese peer para un dado URL*
- *Una conexión TCP no haya fallado con el peer en el último minuto.*
- *El peer ha configurado la opción de **no_query**.*

Si el programa ha habilitado el ***source ping***, envía un mensaje ***ICP_SECHO*** al servidor indicado por el URL. Si este caché no entiende ICP, envía un mensaje ***ICP_DECHO***. Después de haber enviado los mensajes, Squid instala un reloj, para asegurarse de recibir respuesta en un tiempo breve. Normalmente este es de dos segundos.

4.8.3 Proceso de una solicitud ICP

A recibir Squid un mensaje ***ICP_QUERY***, este se procesa de la forma siguiente:

•Se extrae y procesa el URL. Si éste es no válido se devuelve un mensaje ***ICP_INVALID***. Este fenómeno no suele darse dado que el host de origen debería haber certificado la validez del URL.

•Se controlan las listas de acceso locales. Si el acceso es negado, se envía un mensaje ***ICP_DENIED***. La recepción de un mensaje de estos significa error en la configuración de los nodos respectivos.

•Búsqueda local del URL. Si el objeto no existe, o se vencerá en los próximos 30 segundos, retorna un mensaje ***ICP_MISS***.

- Si el objeto es pequeño, se adjunta el mismo como payload de un mensaje ***ICP_HIT_OBJ***

- Si no se cumple ninguna de las reglas anteriores, se responde con un mensaje **ICP_HIT**.

El algoritmo de selección de los peers es simple. Se coleccionan respuestas hasta la recepción de un ***ICP_HIT***, o un ***ICP_MISS***. Inmediatamente de recibido un ***ICP_HIT*** se comienza a descargar el objeto en cuestión. Caso contrario, Squid espera la recepción de las respuestas, o el time-out del reloj que se dispara al realizar la consulta, por un período de tiempo que por default es 2 segundos.

Si un ***ICP_HIT_OBJ*** arriba primero, Squid ha finalizado la transacción y no precisará realizar una conexión HTTP Estos datos recibidos como payload se agregan al caché local.

Si no existen mensajes de HITS, Squid procura traerlos del caché padre. Si no existiera éste, o no respondiera se obtendrá el objeto desde el servidor original.

En general se seleccionan peers, con el RTT más bajo posible. Si bien es razonable su uso para ***ICP_HITS***, su utilización ha sido discutida para la resolución de ***ICP_MISS***s.

Como regla general, se pretende enviar una solicitud a un padre, que esté en la dirección del servidor original.

4.8.4 ICP como detector de inconvenientes

El algoritmo de selección espera el arribo de todas las respuestas, a menos que una de ellas sea un ***ICP_HIT***. Si uno de los nodos se vuelve inalcanzable, los time-out se repiten, con o sin solución de continuidad.

Se designa a un nodo como "muerto" cuando no responde a 20 solicitudes ICP consecutivas. Se continúan los envíos de mensajes ICP a ese nodo, no esperándose respuesta. Cuando una respuesta del mismo arriba se lo marca como "vivo" nuevamente y recomienza el ciclo.

4.8.5 Objetos Públicos y Privados

En las primeras versiones del Caché Harvest, existía una vulnerabilidad, mediante la cual múltiples clientes podían recibir datos de un objeto siendo transferido. Esto permitía que ciertos clientes obtuvieran parte de la información que debería ser exclusivamente de otro.

Este inconveniente se presentaba debido a que Harvest utilizaba el REQNUM del paquete ICP siempre en cero. Squid determina este campo, y lo utiliza para determinar si el objeto es público o privado. Un objeto privado es aquel que puede ser obtenido únicamente por el cliente solicitante; y uno público es pasible de ser accedido por todos.

Todas las solicitudes son iniciadas como privadas, y permanecen así durante la etapa de selección de peer. Después de recibir los encabezados de la respuesta HTTP, el objeto se convierte en público, al menos que se indique lo contrario. Solo los objetos públicos permanecen en el caché, los privados son eliminados inmediatamente después de la transferencia.

4.9 Multicasting

Esta alternativa ha sido propuesta en varios trabajos, como los realizados por Zhang et al.[**13**], como una solución a los inconvenientes de ICP en lo referente a escalabilidad y configuración. Idealmente, multicast puede reducir el tráfico ICP que un caché debe remitir, es decir el número de veces que la aplicación llama a la función *sendto()*; y también puede eliminar los mensajes duplicados por el mismo enlace.

4.9.1 Ventajas e inconvenientes de Multicasting

Sería ideal especificar simplemente un grupo multicast, y que los miembros puedan ingresar o salir de él a voluntad. El Ingresar en un grupo multicast, no requiere tipo alguno de privilegios o autenticación. Este es un problema crítico en el uso de web caché, por lo que si se opta por el uso de multicast, todos los miembros del grupo deben ser identificados y especificados. Multicast asimismo, permite a los otros la captura de las solicitudes ICP, lo que le permitiría recibir una lista de los URLs mas frecuentemente solicitados.

Multicast acarrea problemas asimismo con la restricción de dominios DNS en los peers. Para solucionar esto debería conformarse un subgrupo multicast, los cuales manejarían las restricciones de los dominios establecido en las listas de acceso.

La desventaja final, es la falta de una infraestructura estable y ampliamente diseminada. En algunos casos, como el de nuestra universidad, con escasos recursos económicos que deben ser optimizados de todas las maneras posibles; los cachés web cumplen funciones críticas tales como mejorar el ancho de bando y aumentar la disponibilidad de direcciones IP.

4.10 El CMP (Continuous Multicast Push)

El CMP (Continuous Multicast Push) no posee problemas de consistencia, dado que continuamente envía la información actualizada.

Teniendo en cuenta lo referente a latencia entre los métodos jerárquicos y CMP; podemos decir para multicast, la limitante se encuentra en la velocidad de transmisión entre el servidor y el receptor; mientras que en el caching la limitante se encuentra en la velocidad de transmisión entre el caché mas cercano con el objeto buscado y el cliente que lo solicita. El CMP es el indicado a convertirse en el sistema cuando se deben distribuir objetos populares o rápidamente cambiantes en contenido.

Por lo contrario ¿para qué enviar un objeto constantemente a la red si este no es popular o solicitado frecuentemente?. En este caso es mucho más eficiente la utilización de cachés jerárquicos.

4.11 Proxies para streaming

Los objetos de "streaming" poseen dos diferencias substanciales con los objetos web que por regla general están formados tanto por textos como por imágenes.

- El tamaño de los objetos es normalmente uno o dos ordenes mayor que los objetos web estáticos. Un ejemplo simple se puede ver considerando que una película MPEG-I de una duración de dos horas, precisa 1,4 Gbytes de espacio en disco. Este espacio en disco es finito, por lo que solamente un número de películas podría ser almacenado en un proxy server. Esto disminuye el HIT -rate y la eficiencia general de los sistemas de cacheo. Por supuesto esta película debería ser dividida en segmentos y estos almacenados en servidores proxy.

- Los objetos "streaming" poseen requerimientos de tiempo. Cada segmento posee un tiempo de ejecución para el inicio y finalización, los cuales están interrelacionados. De aquí que cuando la solicitud de streaming arriba al servidor caché, no es simplemente una cuestión de HIT o MISS. Es posible también que la solicitud sea un HIT parcial, donde parte del objeto solicitado está en el caché, y el resto se encuentra en alguna otra parte de la red

Se han propuesto muchas técnicas de para los streaming:

- Segmentación de objetos: Explicitado anteriormente

- Cacheo previo: El proxy almacena los cuadros iniciales de los clips más populares. Después de recibida la solicitud del stream, el proxy inicia la transmisión al cliente, solicitando simultáneamente los cuadros restantes al servidor original.

- Agregado de solicitudes de clientes: Cualquier solicitud diferente del mismo objeto streaming posee una distancia temporal. El proxy sirve tantas solicitudes como pueda desde su memoria o disco.

- Caché dinámico: Las solicitudes de objetos streaming son relacionadas por la distancia temporal. El caché dinámico esconde la distancia temporal entre las solicitudes, de tal manera que con un solo stream de datos puede atender a dos solicitudes. El pathing (encaminado) es utilizado para la obtención de datos perdidos desde servidores originales o servidores caché.

- Cachés cooperativos autoorganizados: La cooperación entre sistemas de caching distribuidos, necesita poseer la capacidad de obtener información de otros cachés y saber de su estado actual referente a información almacenada. Es propuesto asimismo un mecanismo de distribución escalable, en el cual se pude seleccionar el proxy de acuerdo al segmento solicitado por el cliente

4.12 Caching cooperativo

En las configuraciones actuales, es común la existencia de varias redes internas por dependencia/organización, sobre las cuales están instalados varios proxies, generalmente sobre cada punto de acceso a la red. Cada uno de ellos atiende a un número de hosts/paginadores web. Estas máquinas poseen sus propios discos rígidos y su memoria caché. Cuando una solicitud no se encuentra en el caché local (de la máquina) la solicitud es enviada al servidor proxy en la red local. Este revisa que el objeto solicitado esté en su almacén local de objetos, y en caso que no lo posea se contacta directamente con el servidor original.

En este contexto, es mas que relevante considerar la posibilidad de "cooperación entre proxies". Por cooperación entenderemos que si un objeto no es encontrado en el caché de un proxy dado(indicado en el mismo por MISS), se solicitará el servicio de otros cachés de la intranet, antes de solicitarlo al servidor original; ó en el caso de haberse configurado, al caché padre.

Pero aquí deben tenerse en cuenta otros aspectos que juegan un rol muy importante en la cooperación, cuestiones tales como que los objetos deben ser obtenidos tan rápidamente como sea posible, debido a lo cual es preferible muchas veces realizar una cooperación, solamente si existe un beneficio real en su utilización.

4.12.1 La necesidad de cooperación

Entonces surge la pregunta: ¿Porque es necesaria la cooperación? .

El número de ***HITS*** en un proxy caché, depende fundamentalmente de la cantidad de usuarios que comparten el mismo y de la duplicación de solicitudes por ellos efectuadas. Los estudios realizados hasta el momento han demostrado que la utilización de proxies en una población estadística determinada, pueden llegar a reducir las solicitudes remotas desde un tercio a un medio de ellas.

Si bien este número es importante, todas las solicitudes remotas siempre deber remitirse al servidor proxy. Debido a ello la única forma de mejorar el HIT del caché es expandir el uso del mismo a un mayor número de usuarios. Esta es la razón de la utilización de los diseños de caché Web cooperativos.

El overhead del proxy cooperativo se debe a tres funciones del mismo que son: descubrimiento, diseminación y entrega. El descubrimiento tiene que ver cómo el proxy localiza el objeto en el caché. La diseminación es el proceso de selección y almacenamiento de los objetos en el caché, y la entrega define la forma en que el objeto es transferido al cliente desde el servidor original o del caché remoto.

4.12.2 Tipos de Cooperación

En la estructura jerárquica "pura" cada cliente y caché pueden solicitar el objeto de la transacción a solamente otro caché. En la estructura cooperativa, el caché puede obtener el objeto directamente desde varios cachés vecinos. Estos cachés pueden estar dentro de la misma red o no. Los cachés serán hermanos (sibling), si la vecindad se da dentro de la estructura del ISP(empresa o institución). Los cachés además, pueden cooperar también, no estando en la misma estructura.

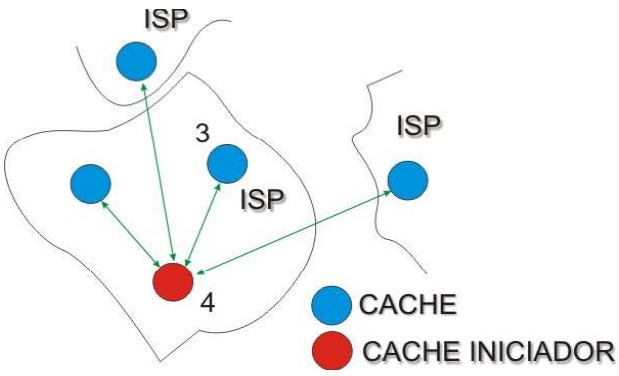

Figura 4.3 Esquema de solicitud entre proxies

Dos de los esquemas más populares de cooperación son los provistos por ICP (Internet Caching Protocol) y CARP (Caché Array Routing Protocol).

Existen diversas formas de configuración para la cooperación entre cachés.

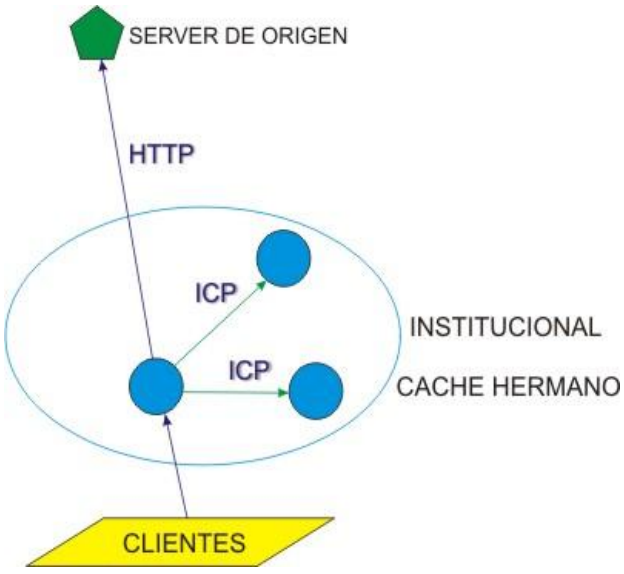

Figura 4.4 Caché cooperativo Institucional/empresarial

Ejemplo de ello sería el caché institucional/empresarial, cuyo mecanismo de funcionamiento quedaría definido como sigue: el usuario contacta el caché institucional/empresarial solicitando un objeto determinado. Si el mismo no lo posee, lo solicita a sus cachés hermanos (sibling) utilizando Internet Caché Protocol. Si alguno de los caché sibling posee el objeto solicitado, el mismo es obtenido y enviado al usuario solicitante. Si ninguno de los siblings lo posee, el mismo es obtenido desde el sitio original por el caché que inicio la solicitud.

Modificando la estructura anterior, podemos instalar un nivel departamental para el caso de empresas y/o instituciones de gran tamaño, las que podrían identificarse con el gráfico siguiente.

Aquí el cliente/usuario contacta al caché departamental, que si no posee el objeto lo solicita a sus cachés hermanos por medio de ICP. Si alguno de ellos lo posee, el caché original lo obtiene y lo envía al usuario en cuestión por medio de HTTP. Si no, contacta al caché institucional vía HTTP.

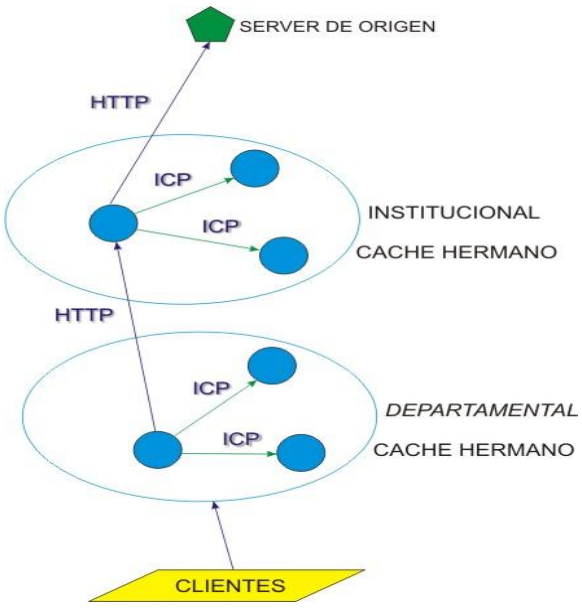

Figura 4.5 Caché cooperativo Institucional/empresarial cooperando en sibling" con un caché departamental

El caché institucional consulta a sus caché siblings por medio de ICP. Si alguno de ellos lo posee, lo obtiene y lo envía al caché departamental. Si así no fuera, el institucional obtiene una copia del servidor original y lo despacha cadena abajo.

Se puede observar claramente la diferencia de funciones entre un caché sibling y un parent. Un hermano, no está en condiciones de resolver un MISS, mientras que un parent sí.

De manera muy especial, se podría configurar el sistema de manera de consultar a todos los cachés en la institución por ICP, a manera de sibling.

En este caso debe tenerse en cuenta que no se debe configurar ICP en los servidores caché en el nivel institucional, dado que a nivel departamental ya realiza la consulta a todos los servidores.

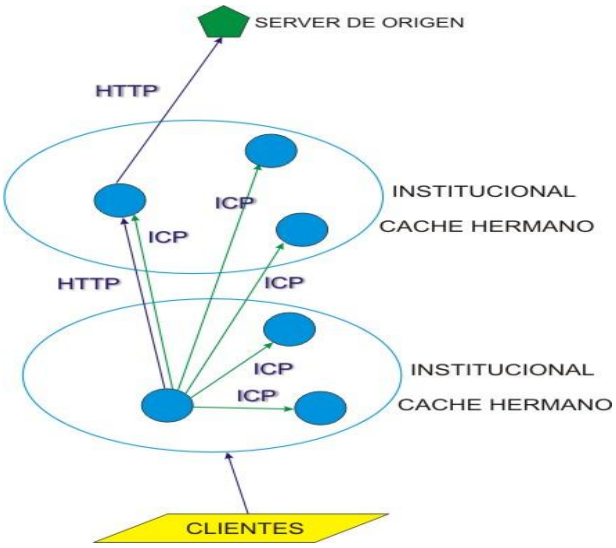

Figura 4.6 Consultas a todos los cachés de la intitución por medio de Internet Cache Protocol

Tomemos el caso de dos ISP donde uno de ellos es dependiente del otro. Estos en general tienen una configuración de peering.

Los cachés hermanos en el ISP dependiente, se consultan a sí mismo utilizando ICP. Los cachés en el ISP dependiente, pueden consultar a mas de un caché en el ISP principal. Los cachés en el ISP principal se consultan así mismos por medio de ICP. Generalmente estos no consultan a los cachés en una jerarquía inferior. Si no poseen el objeto solicitado, lo obtienen directamente de servidor original.

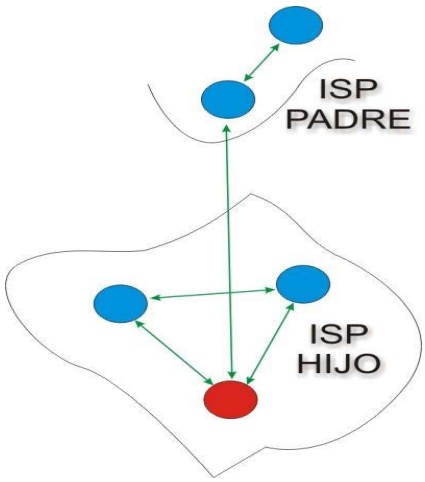

Figura 4.7 Caches de ISP's padre e hijo

Por último quedaría por ver la forma de funcionamiento de los cachés en ámbitos más amplios como serían en el caso de consultas nacionales/internacionales. En este caso dos ISP de distintas naciones son configurados de manera de cooperar entre ellos.

Consideremos que el caché 1 solicita un determinado objeto. Este consulta a sus siblings vía ICP por el objeto. No recibe HIT de los mismos, entonces hace una pedido HTTP al caché 2 vía ICP consulta al 3 y 4. Cuatro responde con un HIT. Entonces caché 2 obtiene el objeto desde caché 4 por HTTP y lo remite por la misma vía al caché 1.

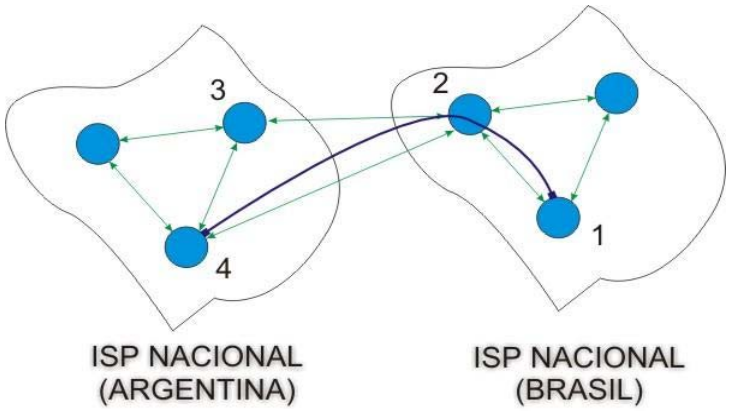

Figura 4.8 Cachés nacionales

4.13 Contenidos distribuidos

4.13.1 CDN y Caching

Las CDN (Redes de distribución de contenidos- Content Distributed Networks) es una tecnología que se está convirtiendo en un rival importante para al caching cooperativo. Krisnamurty et. al.[**16**] reporta que el 1 al 2 % de casi 700 sitios de los más importantes utilizaban CDN en 1999, siendo que en diciembre del 2000 este porcentaje alcanzaba del 17 al 31% de las páginas de los sitios más importantes eran atendidos por CDNs. Como puede observarse es un factor a tener muy en cuenta.

4.13.2 CDN: Funcionamiento

CDN utiliza un método por el cual distribuye réplicas del contenido de un servidor primario a servidores CDN estratégicamente distribuidos en la red, generalmente en puntos de acceso donde concurren los backbones, redes locales y regionales. La solicitud del cliente, es redirigida a servidores CDN. La mayoría de las veces este redireccionamiento se produce en forma transparente a través de entradas en el servicio de Nombres (DNS). Debido a que el CDN controla la replicación y diseminación de objetos, la distribución de contenidos debe considerarse como una replicación de servidores, o como una estrategia de alimentación de contenidos, mas que un esquema de caché sirviendo a clientes.

4.13.3 CDN y Caching, diferencias

CDN difiere del caché cooperativo en otros aspectos. A diferencia de los cachés, la constitución primaria de los CDN son los proveedores de contenidos mas que los usuarios. La relación de HITS tampoco es importante, porque en general se replica el sitio completo, aunque en algunos casos, puede hacerlo solamente de los elementos más activos del mismo. Por ultimo, cualquier usuario puede acceder a un CDN, sin tener que canalizar las solicitudes a través de un proxy.

En el caché cooperativo, el cliente selecciona el sitio, cuando el objeto está disponible en varios lugares. Por lo contrario, la selección del sitio en CDN es hecha directamente por el sistema y es transparente para el cliente. Por estas cuestiones no puede considerarse al CDN como una forma de caché cooperativo.

4.14 Multiplexado

Toda la Red está sufriendo los efectos del protocolo HTTP, dado que en su diseño no se ha tenido en cuenta el comportamiento del protocolo de transporte utilizado, es decir TCP.

HTTP por lo general establece una conexión por cada URI (Uniform Resource Identifiers, Identificador Uniforme de recurso) transferido, con el consabido transporte de los paquetes a ambos lados y los correspondientes RTTs (Round Trip Time). Para objetos pequeños se ha demostrado que estas conexiones TCP tiene un rendimiento muy bajo, debido al mecanismo implícito para el transporte que es el de inicio lento (slow start); como así también debido a los RTTs requeridos para abrir y cerrar cada conexión oportunamente establecida.

Existen tres razones por las cuales la apertura simultánea de varias conexiones TCP se ha vuelto de uso frecuente en el web, a pesar de sus aparentes ineficiencias[**19**]:

1. *Un cliente utilizando éste método percibirá una performance mayor del sistema, debido a que los meta-datos puede ser obtenidos simultáneamente. Con ello el cliente puede formar la página más rápidamente. Clientes que abren múltiples conexiones TCP al mismo servidor, pueden congestionar al mismo en ambiente con enlaces muy cargados, dado que TCP no posee un mecanismo de control de congestión en las aperturas y cierres de conexiones.*

2. *La apertura de varias conexiones TCP, si bien causa problemas de rendimiento en las redes, otorga a quien utilice este método, un ancho de banda mayor respecto al usuario que utiliza una sola conexión TCP. No existe solución para esto a nivel de aplicación, se la debe buscar a nivel de red.*

3. *Para mantener las líneas de bajo ancho de banda y alta latencia*

ocupadas, como las dial-up o satelitales, dado que el inicio lento podría producir un desperdicio de ancho de banda.

4.14.1 El tráfico generado por TCP

La diferencia fundamental entre el HTTP/1.0 y HTTP/1.1 es la forma en que maneja el primero de ellos las conexiones TCP persistentes (Keep-Alive). Estas no fueron diseñadas para trabajo conjunto con proxies, y no atiende solicitudes de pipelining (entubamiento). El segundo de ellos, cumple estas funciones en modo normal, sin ninguna característica de configuración.

La conexión persistente y el pipelining reducen el tráfico en la red y especialmente el número de mensajes TCP necesarios para el establecimiento y el cierre de una conexión.

Estos inconvenientes son propios de los ambientes donde el uso de HTTP es el mayoritario. Puede tratarse de prevenir y/o solucionar los presentados con dos acciones fundamentales:

- Activar el Radom Early Detection (RED) u otros algoritmos activos de control de congestión en los routers, para poder asegurar el ancho de banda igualitario a todos sus clientes cuando la red se encuentre congestionada.

- Desarrollar y aplicar protocolos de multiplexado para el uso de HTTP, y eventualmente otros protocolos. De este modo múltiples objetos pueden ser obtenidos desde un servidor en el web con una sola conexión TCP.

4.14.2 Experiencias sobre el tema

La idea de la utilización del multiplexado para las conexiones TCP, se ha planteado en numerosas publicaciones [**18,19,20**] a lo largo de los últimos años, fundamentalmente en trabajos teniendo que ver con enlaces de alto delay como los SCPC[15] satelitales.

El multiplexado de sesiones de distintos protocolos sobre una misma conexión TCP beneficia el comportamiento de la red, porque obviaría el slow-start típico del protocolo TCP para cada conexión, reduciría el overhead sobre los end-points y por ello utilizaría los recursos de la red en una forma más racional.

Mucho se trabajó en esto sobre enlaces satelitales, y otro tipo de topología de redes con alto retardo debido al tamaño, equipamiento y/o complejidad (Long Fat Networks). Estos trabajos se han referido casi siempre al overhead correspondiente a las conexiones TCP.

El inconveniente principal en el uso del muxing[16] se presenta cuando estos protocolos deben competir con todo el tráfico extra existente en la red como background.

Mientras que los beneficios del multiplexado fueron bien entendidos en modelos exponenciales de tráfico; el impacto de su funcionamiento en redes de importancia por su tráfico y gran extensión; están siendo tomado en cuenta recién en trabajos de los últimos años.

La línea de base del modelo de multiplexación, en un simple servidor de colas, de buffer finito, bajo la forma de atención FIFO[17].

4.14.3 WebMUX y SCP[18]

El crecimiento en cantidad de los documentos con contenido multimedial en los distintos servidores alrededor del planeta, se ha convertido en un inconveniente en el núcleo de las redes de comunicaciones. Fundamentalmente debido a la necesidad de cambio de paradigma que asignaba como función primaria a las comunicaciones tradicionales la transmisión de la información; y donde la calidad el servicio se lograba con un mayor ancho de banda y/ó con mejores equipos de conmutación.

Sin embargo, en redes donde el almacenamiento de la información juega un rol fundamental; un mejoramiento en la calidad de servicio es impensable sin la utilización de otras herramientas como el caching, el pregrabado y/o contenido distribuido de la información.

Uno de los esfuerzos en este sentido es la aparición del protocolo de multiplexado WebMUX, el cual fue diseñado para multiplexar conexiones TCP bajo HTTP, de tal manera que este último no deba cambiar en su comportamiento, permitiendo así mismo la coexistencia de múltiples protocolos, por ejemplo HTTP, HTTP/NG, etc, el cual hará mucho mas fácil la migración a protocolos web del futuro como así también la comunicación con los applets de los clientes usando protocolos privados, sobre la misma conexión TCP de la conversación HTTP.

Algunos de los objetivos que se pretenden de WebMUX, siendo utilizado como protocolo de multiplexado en el Web, son: Servicio sin confirmación y sin negociación o round trips al servidor; diseño simple, elevada performance, libre de bloqueos y que permita que múltiples protocolos sean multiplexados sobre la misma conexión TCP.

---

[15] Single Carrier per Channel
[16] Multiplexado
[17] Primero en llegar, primero en atender; First In First Out
[18] Session Control Protocol

El multiplexado de múltiples sesiones sobre una misma conexión TCP introduce una causa potencial de bloqueo mutuo, que este protocolo pretende obviar. Se introduce con WebMUX el concepto de "garantía". El receptor garantiza al transmisor que una cierta cantidad de espacio de buffer esta garantizado para recibir su transmisión y el transmisor garantiza que no transmitirá mas que esa cantidad. Si se cumple este arreglo, no existe bloqueo mutuo.

WebMUX fue pensado y diseñado para su utilización entre un cliente final y un servidor web o proxy. Este protocolo se convierte en una alternativa a las conexiones TCP persistentes, y su implementación en principio fue apuntada a la utilización en conjunto con equipos de bajo ancho de banda, por ejemplo módems celulares.

El control de flujo realizado por este protocolo es muy rígido, con los inconvenientes que ello acarrea. Utilizando WebMUX los proxies experimentan cambios dinámicos del número de conexiones concurrentes al servidor.

Otro protocolo similar, con las mismas funciones que el WebMUX es el SCP [**22**]. Dado SCP es un poco más antiguo WebMUX, posee ciertas limitaciones, como la posibilidad de bloqueo si no existe memoria ilimitada, pero sobre todo un overhead de 8 bytes que se considera demasiado para las funciones a cumplir.

4.14.4 Tunelizado a Nivel de Aplicación

Aparte de los inconvenientes suscitados por el costo de apertura y cierre de las conexiones TCP, existen determinadas rutas que necesariamente deben pasar por grandes redes saturadas de tráfico de todo tipo. A estas se las llaman LFN[19].

[19] Long Fat Networks, Grandes redes pesadas

Recientes trabajos se han realizado utilizando tunelizado (tunneling) de datos, no de paquetes, para mejorar la latencia y transferencia de contenidos, proceso al que se ha dado en llamar tunelizado a nivel de aplicación (Aplication Level Tunneling) [**21**]

El RTT (Tiempo de respuesta) es la constante que utiliza el TCP para su retroalimentación. Si este valor es elevado se produce una respuesta muy lenta a las condiciones del enlace, ocurriendo oscilaciones y desequilibrios en el tráfico; por lo que enviando mas paquetes, se pierden muchos de ellos y existen muchos reenvíos.

Estos inconvenientes son propios de los escenarios donde existen vínculos muy diferenciados y heterogéneos, por los cuales indefectiblemente debe atravesar una ruta determinada. Debido a los elevados retardos, una conexión TCP end-to-end se extinguiría. La solución encontrada fue utilizar un proxy para la conexión sobre la capa de red, para que las pérdidas sean aisladas en cada una de las redes buscándose un alto throughput y disminución de la latencia.

Una de las soluciones se planteó con la utilización de servidores proxy-caché en los límites de cada red, tunelizando las sesiones a través de las redes en tránsito. Con esto se busca bajar la latencia, lo que a su vez significa tener poca carga en los servidores proxy.

El proceso de multiplexado empieza con el diseño del protocolo a ser utilizado. Este debe comportarse como un TCP normal, con alguna variante que permita el intercambio de mensajes, fichas, etc; con lo cual se simplifica en ambos extremos el manejo de los bufferes y de las colas. Una vez que se logrado la modificación del protocolo TCP con el propósito anterior, se buscó la interacción de este y de un servidor conocido en Internet como lo es el Squid.

El comportamiento del protocolo en la apertura de la conexión y liberación de la misma; como así también la forma de cierre de la conexión por inconvenientes en el transporte puede verse en la siguientes figuras presentadas aquí.

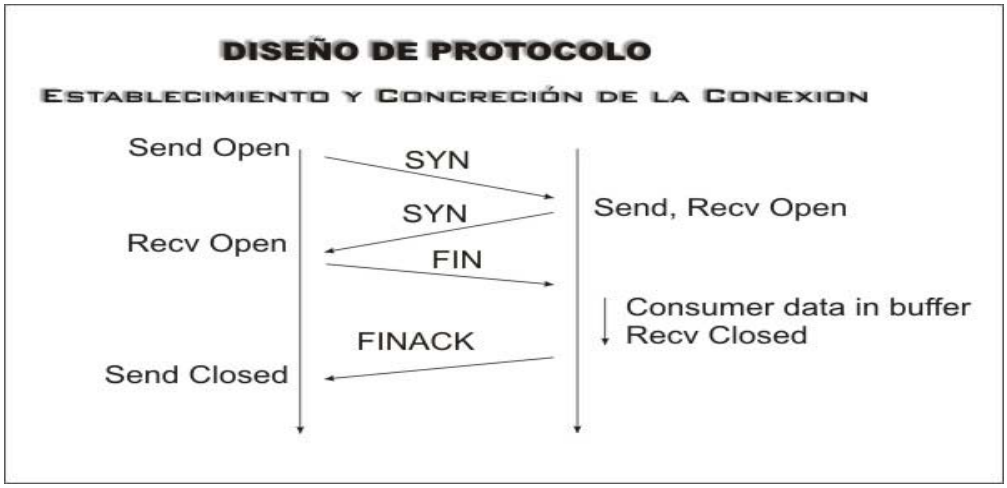

Figura 4.9 Variante de protocolo TCP adaptado a muxing

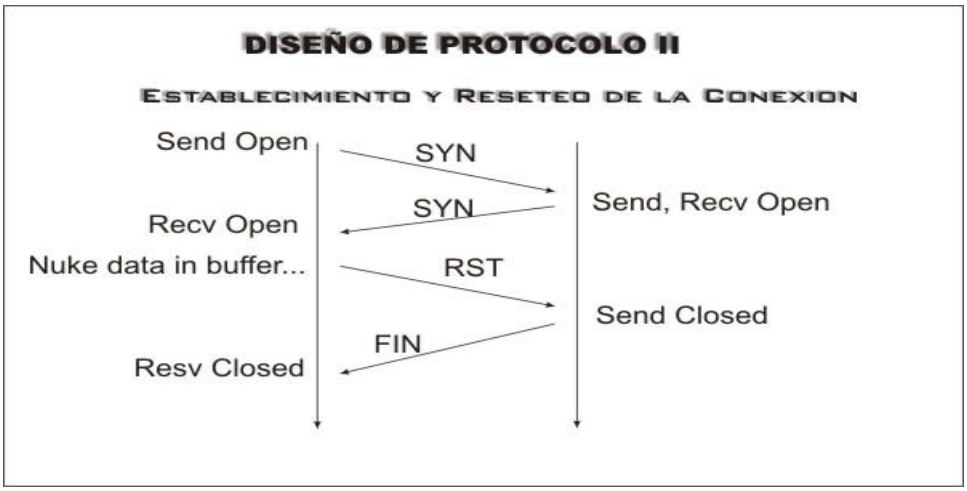

Figura 4.10 Establecimiento y reseteo de la conexión en el proceso *de Multiplexado*

En ambos gráficos puede observarse que el protocolo debe tener en cuenta en todo momento la existencia de buffers para el funcionamiento del mismo.

Dado que TCP es un sistema de transporte confiable, no se puede dejar caer la conexión simplemente porque el espacio de buffer se agote. Por ello el receptor debe

reservar un espacio de memoria para almacenamiento de datos, información que debe ser de alguna forma comunicada al otro punto de la comunicación. Las pequeñas sesiones de conexión y desconexión, son solapadas por los grandes intervalos durante los cuales se envían datos.

El mismo escenario se presenta cuando un servidor rápido, debe atender a clientes muchos más lentos.

El sistema para la reserva de buffer basado en el sistema de fichas, se pensó de la manera siguiente. Cada enlace posee una cantidad de memoria previamente fijado, el cual es representado por fichas. En el saludo inicial, todas las fichas son enviadas al peer. Con esto se establece cuanta información puede ser transferida en la sesión. Cuando el buffer es liberado, se indica con un ACK, para la sesión indicada.

Otro método utilizado debe reaccionar con la dinámica de la sesión. Dos formas son posibles, compartida y por intuición. La compartida se calcula por la ecuación determinada por: buffer total dividido por 2 veces el números de sesiones activas menos el buffer utilizado. El intuitivo toma la mitad del buffer, lo divide por la forma compartida definida anteriormente, asignando a las comunicaciones existentes la cantidad compartida; el 50 % remanente de memoria se asigna para nuevos arribos que pudieran hacerse.

Existen diversas variantes sobre la metodología de asignación de buffers. Como dijéramos anteriormente, las sesiones están en una cola round-robin, cada una de ellas regida por la asignación previa de bufferes.

Todo esto se combina con el paquete Squid, para ser utilizado en la periferia de las redes LFN.

Este paquete provee un diseño no bloqueante, el programador (scheduler) maneja lo correspondiente a la apertura y timers de conexiones. A su vez el módulo de comunicaciones posee un sistema no bloqueante de llamado a los sockets. La

tarea desarrollada en el trabajo de referencia, tiene que ver fundamentalmente con la multiplexación de los tentáculos del paquete Squid.

Las conclusiones de este trabajo establecen que la utilización de Squid multiplexado sería muy conveniente en ausencia de otro tipo de tráfico. Una sola conexión sería el mejor escenario; pero se complicaría el panorama si esta se cae. Es contraproductivo en el caso de múltiples conexiones, y lo definitivamente comprobado es que depende del tráfico existente en las redes.

Lo que también queda claro en estos trabajos es que la multiplexación reduce la latencia y el retardo. Contrariamente a lo anterior, no queda suficientemente establecido es que sea aceptable el throughput, dado que las conexiones establecidas por un período prolongado experimentan el fenómeno de descompensación (overshoot). Las conexiones por corto tiempo no producen este fenómeno si analizamos el throughput individual, pero no se ha determinado fehacientemente el efecto de los trhroughput individuales.

Más importante aún, en estos trabajos destacan que el multiplexado es potencialmente muy bueno para ser aplicado como base en Tunelizado a nivel de aplicación(Application Level Tunneling) y que si bien este es el indicado para enlaces satelitales u otros tipo de vínculo que unen puntos distantes, puede tener aplicación en otros ambientes como VPNs.

Cuando se busca aplicaciones similares en proveedores y/o Internet, se percata uno de la inexistencia de estas soluciones, debiéndose principalmente a que el protocolo no se ha aplicado en forma robusta; y sobre todo, no forma parte en la totalidad del paquete standard Squid. Debido al hecho anterior, es Squid que fuera utilizado en la experiencia referida, fue acondicionado de manera que su programador de colas realizara su tarea "al tanteo", en lugar de realizar una estimación del ancho de banda para optimizar su tarea.

CAPÍTULO V: Resultados

5.1 Origen de los datos

Los resultados presentados en esta instancia, se corresponden con los publicados en las páginas web de cada uno de los servidores caché de la Universidad de Karlsruhe.

La política de esta Universidad, no permite la utilización de los archivos de registro si los mismo no están anonimizados, en cuanto al origen y destino de la solicitud realizada al servidor proxy. Solicitar la autorización correspondiente a las autoridades universitarias, a fin de contar con los archivos originales, hubiera demandado un tiempo no disponible de permanencia extra en la República Alemana.

Aparte de las cuestiones de forma anteriores, existe un impedimento físico, y que tiene que ver directamente con la capacidad de almacenamiento de archivos generadores de la información. El mantenimiento de los archivos de referencia para su procesamiento supuesto semanalmente agotaría la capacidad de almacenamiento de los equipos en 4,5 días de acuerdo a la cantidad de solicitudes por hora recibidas.

Para la generación de la información se utilizó el paquete de libre distribución Squeezer[20] [**32**].

Los reportes fueron publicados, y aún se mantienen[21] en formato hipertexto, en los siguientes sitios:

[20] Configurado en forma conjunta con los administradores.
[21] al 31 de julio de 2002

Tabla 5.1 Servidores y sus URL's en la Universidad de Karlsruhe

proxy1.rz.uni-karlsruhe.de/stats	129.13.64.201
proxy2.rz. uni-karlsruhe.de/stats	129.13.64.202
proxy3.rz. uni-karlsruhe.de/stats	129.13.64.203
proxy4.rz. uni-karlsruhe.de/stats	129.13.64.204

5.2 Período de la Experiencia

El presente trabajo se ha realizado desde el mes de marzo hasta el mes de junio próximo pasado, durante una visita de estudio a la Universidad de Karlsruhe financiada por el DAAD[22] y propiciada por una invitación de la Dirección del Instituto de Telemática de la Facultad de Ciencias de la Computación[23].

Si bien la permanencia en el Instituto se extendió desde el 15 de enero hasta el 12 de abril, los datos validados son aquellos comprendidos entre el período de marzo a junio de 2002.

5.3. Resultados obtenidos

Para tener una idea acabada del tamaño de la población atendida, cabe acotar que el número de solicitudes atendidas por los cuatro servidores durante el período de permanencia en el Instituto de Telemática alcanzó una cifra cercana a los 550

[22] Deutscher Akademischer Austauschdienst, Servicio Alemán de Intercambio Académico.
[23] http://www.tm.uka.de/

millones. El promedio alcanzado con este valor alcanza a 45.110 requerimientos por hora y servidor instalado.

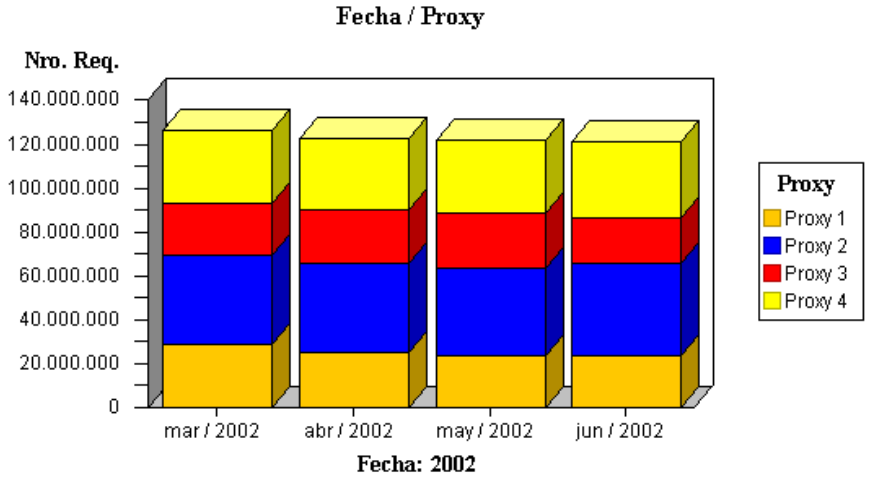

Figura 5.1 Requerimientos caché atendidos por los servidores de la UKA, acumulativos, mensuales

La figura 5.1 fué generada por una aplicación que procesara los reportes diarios generados por el tándem Squid-Squeezer. Para poder observar una muestra de los reportes referirse a los Anexos 1 al 5. Se puede observar que la cantidad de requerimientos es similar en cada uno de los períodos y referentes a los cachés.

El valor de 45.110 solicitudes por hora mencionado anteriormente, no incluye los resultados obtenidos durante el período de puesta a punto tanto de servidores Caché como de los generadores de reportes. Estas tareas se realizaron durante parte del mes de Febrero.

En la composición de las requisitorias de servicio efectuadas por los clientes en el período de observación, existe marcada similitud en la conformación de las distintas columnas, en lo referente al número de operaciones realizadas. Si a esto adicionamos lo observado en la representación gráfica del tráfico generado por estos “requests”; se aprecia el buen funcionamiento de la opción “balance de carga” en las

aplicaciones utilizadas. El tráfico, alcanza aproximadamente los 1,5 Terabytes para cada uno de los servidores instalados, equivalente a un tráfico horario de 550 Megabytes por servidor como se observa en la figura 5.2.

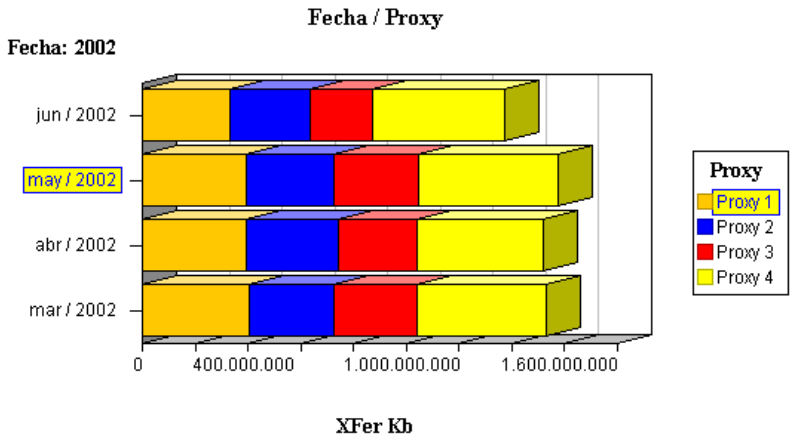

Figura 5.2 Tráfico generado en los servidores caché por atención de solicitudes

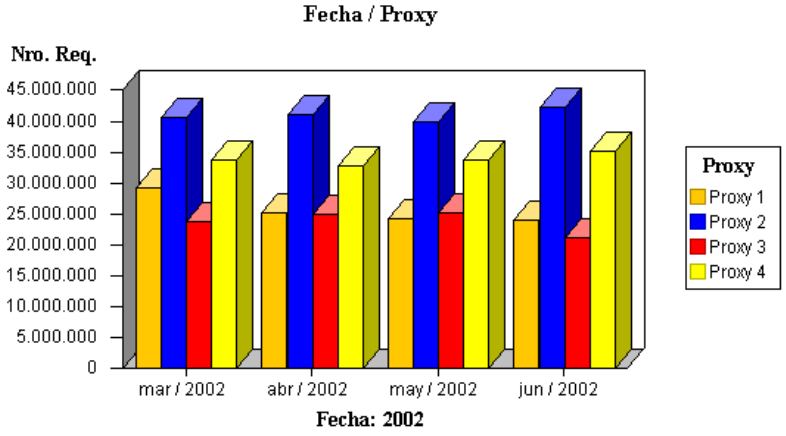

Figura 5.3 Solicitudes mensuales discriminadas por servidor

En las Figuras 5.3 y 5.4 se puede observar con más detalle la discriminación por servidor de las solicitudes mensuales y el tráfico generado por éstas. Esta representación gráfica nos presenta una primera idea del tamaño, cuando no referimos con relación a otras instalaciones. El factor aproximado de relación de tamaños es de 30, con respecto a de la Universidad de Misiones.

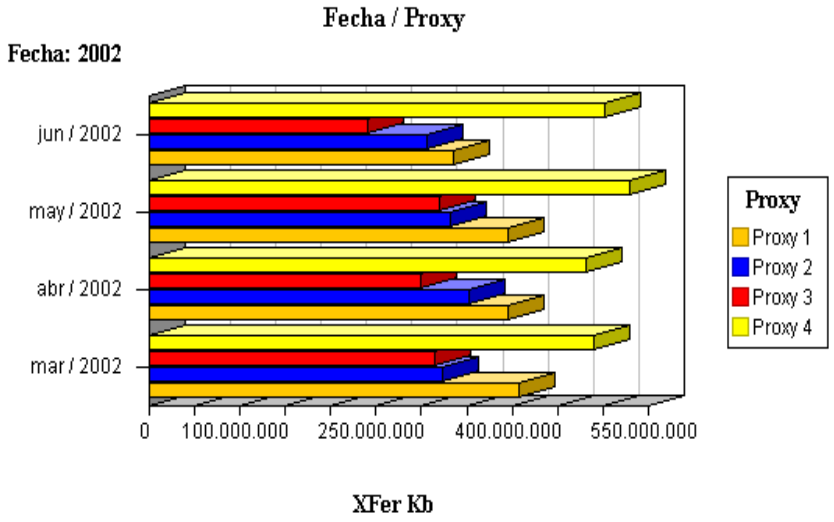

Figura 5.4

5.4 Caracterización de la carga de los servidores

Para poder caracterizar el comportamiento y la carga de los servidores caché, en necesario en primer lugar realizar una caracterización del tráfico hacia estos, y con ello determinar las condiciones esenciales y primordiales para el buen funcionamineto de los equipos, tanto con referencia a las redes como al protocolo HTTP in sí. Un caché si está bien diseñado, debe ser capaz de soportar la carga total presente en la red.

Los diseños de cache son planificados y proyectados generalmente con el uso de simulaciones. Estas simulaciones pueden correrse a partir de datos reales generados por un servidor de existencia física, o los datos necesarios para el diseño de los servidores son generados por funciones analíticas.

La caracterización de los Servidores WWW ha sido publicada por Almeida et al [**29**], Arlitt et al [**30**], Bestavros et al [**31**] y otros autores en distintas ocasiones. El estudio de Arlitt y Williamson, identificó características propias, que también son

llamadas invariantes, comunes a archivos de registro en los servidores utilizados en los estudios.

Aparte del establecimiento de estas invariantes, estos autores pudieron establecer que el tamaño de los objetos solicitados en un servidor WWW posee una distribución que se corresponde con Pareto, y que los intervalos entre solicitudes de servicio poseen una distribución exponencial.

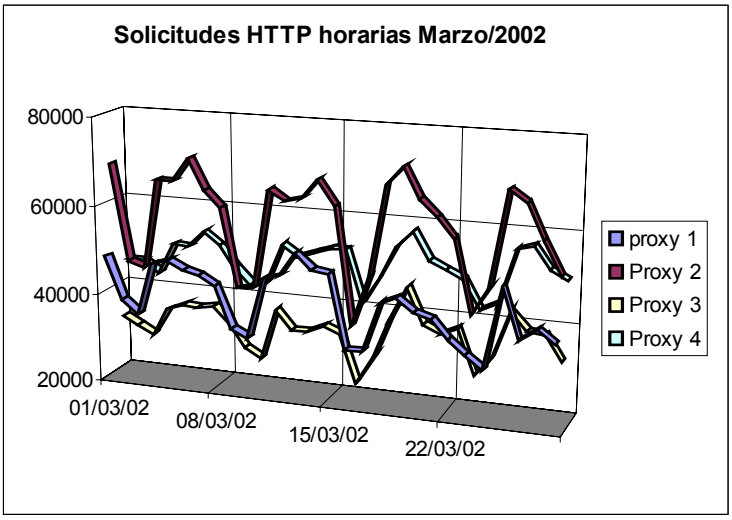

Figura 5.5 Solicitudes/hora.día para el mes de marzo 2002

A fin de poder realizar estudios de similitud con estos mencionados anteriormente, y para no incurrir en la generación de un gráfico engorroso que no muestra información útil, se han elaborado los gráficos de solicitudes por proxy por hora y por día, discriminado para los cuatro meses de estudio en los gráficos 5.5 a 5.8.

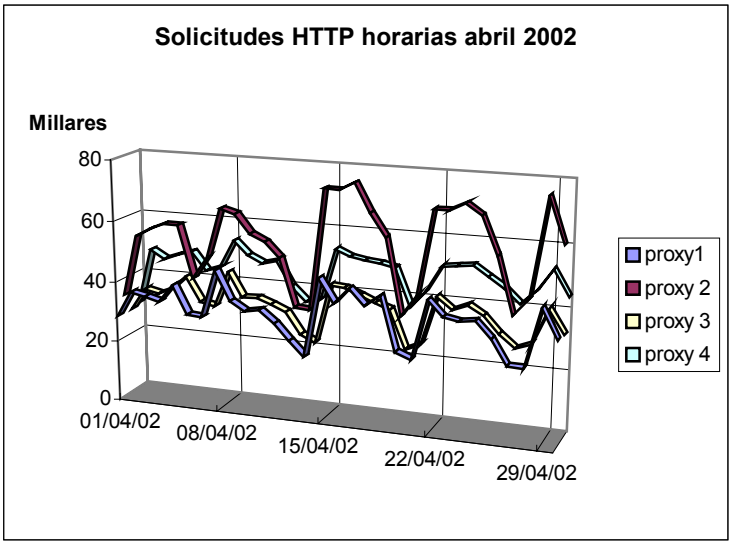

Figura 5.6 Solicitudes/hora.día para el mes de abril 2002

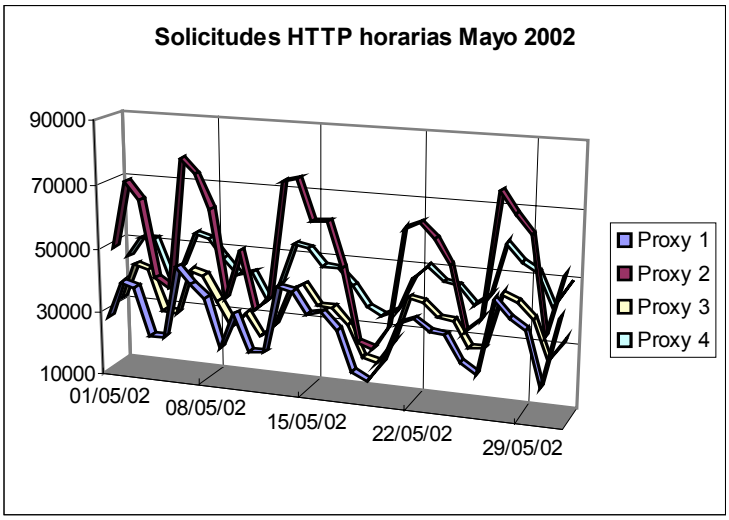

Figura 5.7 Solicitudes/hora.día para el mes de mayo 2002

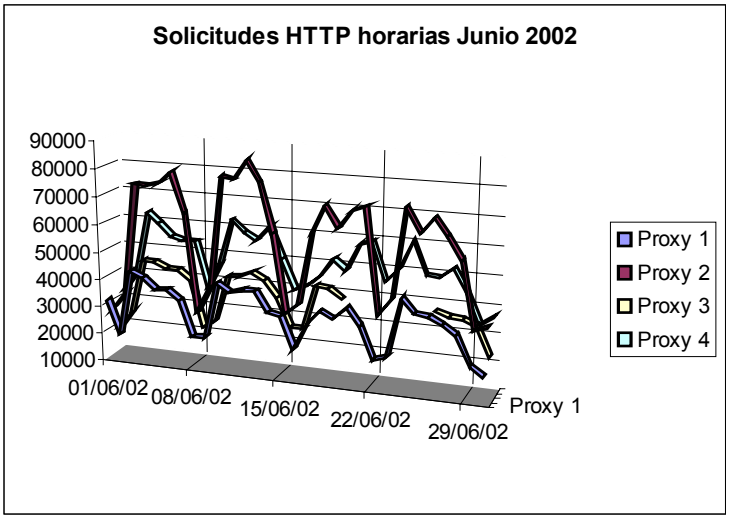

Figura 5.8 Solicitudes/hora.día para el mes de junio 2002

En los gráficos presentados anteriormente, puede observarse el comportamiento similar a lo largo del eje de tiempo, donde a cada incremento del valor de solicitudes en uno de los servidores, es decir en las ordenadas, le corresponde con un aumento proporcional en el valor para los otros 3.

Para evaluar y/o establecer la correspondencia entre el número de solicitudes y el volumen transferido desde los servidores por los request graficados anteriormente, a un aumento en el número de solicitudes, debería corresponderse con un aumento en el volumen de tráfico diario. Los gráficos 5.9 a 5.12 representan la transferencia desde cada uno de los servidores proxy. En el gráfico 5.12 puede observarse un período de 6 días en los cuales no estuvo en funcionamiento el proxy número 3, representado como una interrupción en gráfico respectivo

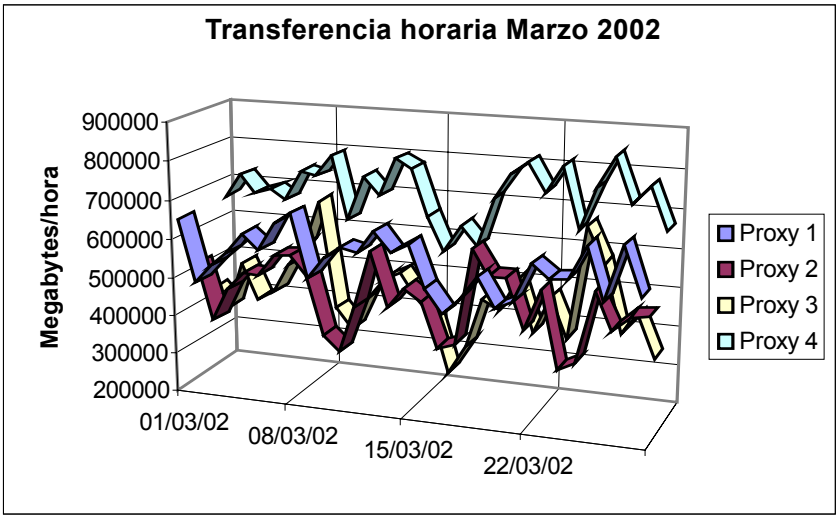

Figura 5.9 Transferencia horaria por servidor mes de marzo 2002

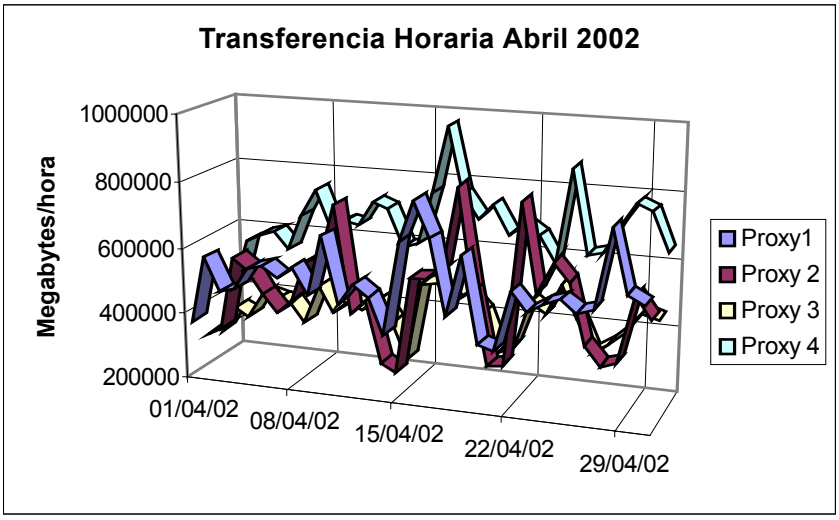

Figura 5.10 Transferencia horaria por servidor mes de abril 2002

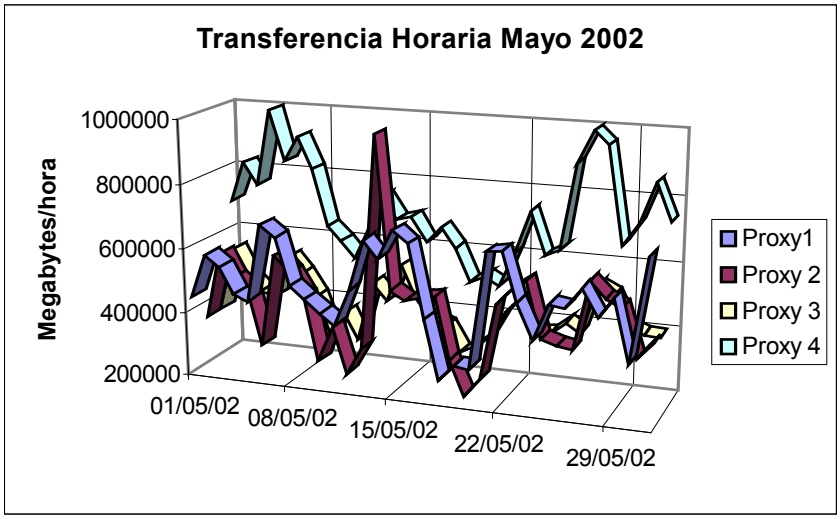

Figura 5.11 Transferencia horaria por servidor mes de mayo 2002

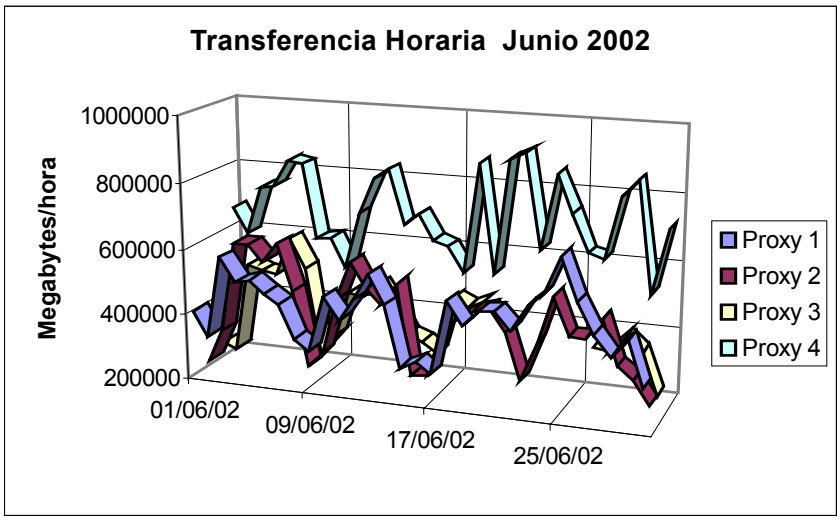

Figura 5.12 Transferencia horaria por servidor mes de junio 2002

De las ilustraciones anteriores, no es posible visualizar una relación directa entre solicitudes y volumen transferido.

5.5 Caracterización del tipo de tráfico

Por lo dicho en el párrafo anterior, es importante establecer la importancia de cada tipo de archivo en el volumen total de transferencia. Estos archivos son solicitados directamente, o puedes ser enlazados desde aplicaciones hipertexto.

Muchos trabajos se han realizado sobre simulaciones respecto a protocolos de cacheo, en los cuales se ha tenido en cuenta solamente la carga de los servidores y no se prestó suficiente atención a los tiempos de respuesta. Esto es una obviedad dado que es difícil modelar en Internet y porque no existen en ese entorno datos de HTTP que sean globalmente dependientes del tiempo.

En el sistema analizado, se han establecido los tipos de archivos que generan el 99% de las solicitudes a los distintos servidores. Estos valores han sido obtenidos del procesamiento de los reportes de funcionamiento publicados por el RZ[24] en cada uno de los servidores. Como ejemplo de los reportes referidos, se han añadido los anexos 2, 3 ,4 y 5 donde puede observarse un reporte tomado como modelo, que fuera emitido respecto a los valores reportados por los diferentes servidores el día 01 de Marzo de 2002.

Es importante representar gráficas que determinen esta recurrencia respecto al tipo de archivo solicitado. En el gráfico 5.13 se pueden observar las solicitudes efectuadas por tipo de archivos, los cuales son causantes del 99% de todas las requisitorias.

En los 6 primeros lugares de la tabla 5.2 podemos observar que son ocupados por archivos de imágenes, texto y aplicaciones como un porcentaje importante por un tipo de archivo no definido en los estándares.

---

[24] Rechenzentrum, Centro de Cómputos

Tabla 5.2 Tipos de archivo causantes del 99% de las solicitudes a los servidores caché

	Proxy 1	Proxy 2	Proxy 3	Proxy 4
image/gif	*52.485.646*	*86.999.802*	*38.368.809*	*49.535.692*
text/html	*23.059.023*	*38.556.213*	*20.869.275*	*36.819.442*
image/jpeg	*16.488.555*	*22.238.937*	*20.475.574*	*28.610.314*
sin definición	*16.438.773*	*22.117.268*	*14.720.656*	*22.357.410*
application/x-javascript	*3.155.134*	*5.670.778*	*3.509.427*	*5.162.571*
text/plain	*1.845.482*	*5.045.136*	*1.818.904*	*3.841.134*
text/css	*1.288.469*	*2.730.759*	*937.730*	*1.165.466*
image/png	*571.961*	*575.267*	*232.164*	*435.468*
application/octet-stream	*526.615*	*507.129*	*915.091*	*1.282.648*
application/x-shockwave-flash	*447.201*	*668.187*	*347.652*	*543.312*
video/mpeg	*161.372*		*252.911*	*301.576*
text/xml			*258.795*	*478.766*

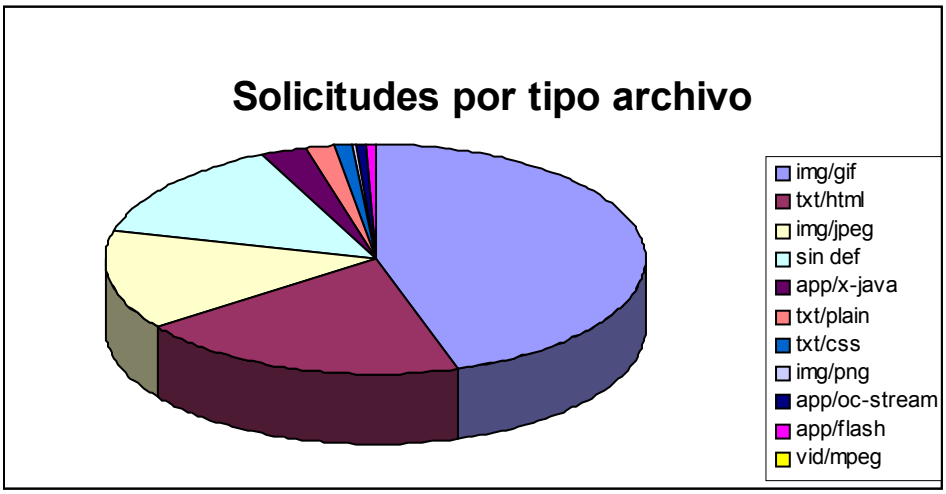

Figura 5.13 Distribución de los tipos de archivos causantes del 99 % de las solicitudes

Los archivo de gráficos tipo GIF y JPG representan un 57% de las solicitudes, alcanzando una cifra cercana a los 315 millones de HITS, mientras que el valor alcanzado por los archivos de texto, HTML y plano, alcanzaron un poco mas del 24%.

Para efectuar una primera observación de la relación entre el valor de HITS y el tráfico generado por estos, se han tabulado los valores de tráfico en la Tabla 5.3.

Debido a la gran dispersión de los datos obtenidos del procesamiento de los reportes generados por el Squeezer, se han seleccionado las entradas que son responsables del 95% del tráfico total en los proxies durante el cuatrimestre comprendido entre los meses Marzo-Julio. Todos estos valores están expresados en Kilobytes.

Tabla 5.3 Transferencia por tipo de archivo; en Kilobytes, responsables del 95% del tráfico en los servidores caché

MIME Type/proxy	Proxy 1	Proxy 2	Proxy 3	Proxy4	Total	%
Application/octet-stream	288.065.416	172.175.920	232.498.313	327.052.832	1.019.792.481	15,85
text/plain	208.004.480	139.052.723	89.582.436	303.962.673	740.602.312	11,51
text/html	180.927.163	361.116.059	165.168.309	299.764.168	1.006.975.699	15,65
image/jpeg	170.716.096	198.301.136	184.520.505	285.888.561	839.426.298	13,04
video/x-msvideo	148.042.761		94.936.910	175.844.730	418.824.401	6,51
image/gif	118.263.128	167.636.509	94.391.654	128.708.716	509.000.007	7,91
video/mpeg	103.537.354	48.641.364	109.992.262	161.023.615	423.194.595	6,58
application/zip	86.700.459	53.268.780	53.213.782	106.630.501	299.813.522	4,66
audio/mpeg	72.300.318	19.945.912	55.368.774	79.536.796	227.151.800	3,53
sin definición	51.611.570	60.817.361	23.287.662	68.338.053	204.054.646	3,17
Application/x-zip-compressed	42.331.333	24.213.999	28.815.700	53.621.594	148.982.626	2,32
audio/x-pn-realaudio	33.720.239	9.454.315	17.944.534	28.970.316	90.089.404	1,40
video/msvideo	23.363.800	57.250.483		8.335.443	88.949.726	1,38
video/x-ms-asf	21.863.021		6.259.290	13.188.351	41.310.662	0,64
Application/vnd.rn-realmedia	19.878.279				19.878.279	0,31
video/quicktime	17.636.618	12.475.405	12.093.867	23.799.605	66.005.495	1,03
application/pdf	17.309.893	25.342.971	10.321.999	18.069.036	71.043.899	1,10
Application/x-shockwave-flash	11.489.909	15.395.179	10.000.098	17.526.184	54.411.370	0,85
application/x-tar	11.289.641	13.733.757	7.127.157	12.385.621	44.536.176	0,69
Application/x-debian-package	9.715.963			9.766.337	19.482.300	0,30
video/x-ms-wmv	7.946.468			8.191.227	16.137.695	0,25
Application/x-javascript	6.747.503	12.311.079	8.934.171	12.198.627	40.191.380	0,62
audio/mp3	5.700.177				5.700.177	0,09
multipart/byteranges		9.877.380			9.877.380	0,15
Application/download		8.276.978			8.276.978	0,13
Application/x-msdos-program		8.056.534			8.056.534	0,13
Audio/x-mpeg			8.186.134		8.186.134	0,13
Application/x-mpeg			5.086.176		5.086.176	0,08
Totales					6.435.038.152	100

En ésta tabla se observa que las aplicaciones ocupan un lugar primordial cuando analizamos el tráfico en las redes, alcanzando casi un 16% del total de 6,5 Terabytes. Los archivos de texto, presentan una relación lineal con las solicitudes, dado que HTML y plano alcanzan al 26%. La transferencia de gráficos alcanza al 21 % y existe un componente multimedial que alcanza aproximadamente al 17% para los archivos de audio y video (x-msvideo,audio y video mpeg).

5.6 Rendimiento de la Jerarquía de Caches

A lo largo de los cuatro meses se ha encontrado un porcentaje de transferencia en disco que alcanzó un promedio del 13,54%, y con las solicitudes a los proxies vecinos que se establecieron con Siblings-hits 1,13%.

El tabla siguiente muestra los valores promedio obtenidos por servidor considerado. También se muestran los intervalos de valores obtenidos

Tabla 5.4 Rendimiento de la jerarquía de cachés en la UKA

	Proxy 1	**Proxy 2**	**Proxy 3**	**Proxy 4**
% de transferencia desde caché propio	*14,03*	*13,59*	*16,30*	*10,25*
Intervalo de valores en servidor local %	*7,42-20,7*	*7,50-18,59*	*9,66-23,37*	*4,96-16,44*
Otros servidores (prom %)	*0,86*	*0,79*	*1,23*	*1,64*
Intervalo de valores otros servidores en %	*0,34-2,08*	*0,26-2,04*	*0,54-2,47*	*0,67-3,04*
Total de Jerarquía	*14,88*	*14,38*	*17,53*	*11,89*

5.7 Ajuste de Datos usando Regresion No Lineal

Se ha elaborado un modelo estadístico, que considera a las siguientes variables:

ttnroreq: Requerimientos totales recibidos en Hits

xfer_kb : Transferencia total efectuada en Kilobytes

dfnroreq: Requerimientos a servidores originales

dfxferkb : Transferencia desde servidores originales

tsnroreq: Requerimientos a servidor local Hits

tsxferkb: Transferencia desde el servidor local

Se ha considerado ttnroreq, como variable dependiente y las demás consideradas independientes.-

Se ha analizado la contribución de xfer_kb y dfxferkb a la variable dependiente, como contribuciones parciales a un modelo de regresión múltiple.

A fin de poder determinar que tan bien predice la variable independiente en el modelo estadístico, se necesita desarrollar varias medidas de variación.

La Variación Total mide la variación de Y en torno a su media YM.-

A su vez esta variación se puede dividir en dos grupos, variación explicada (atribuible a la relación X e Y), y variación no explicada, atribuida a factores desconocidos (llamados generalmente ruidos). Se considera a la Transferencia total en Kilobytes como variable Independiente, y al Total de requerimientos en hits variable dependiente.

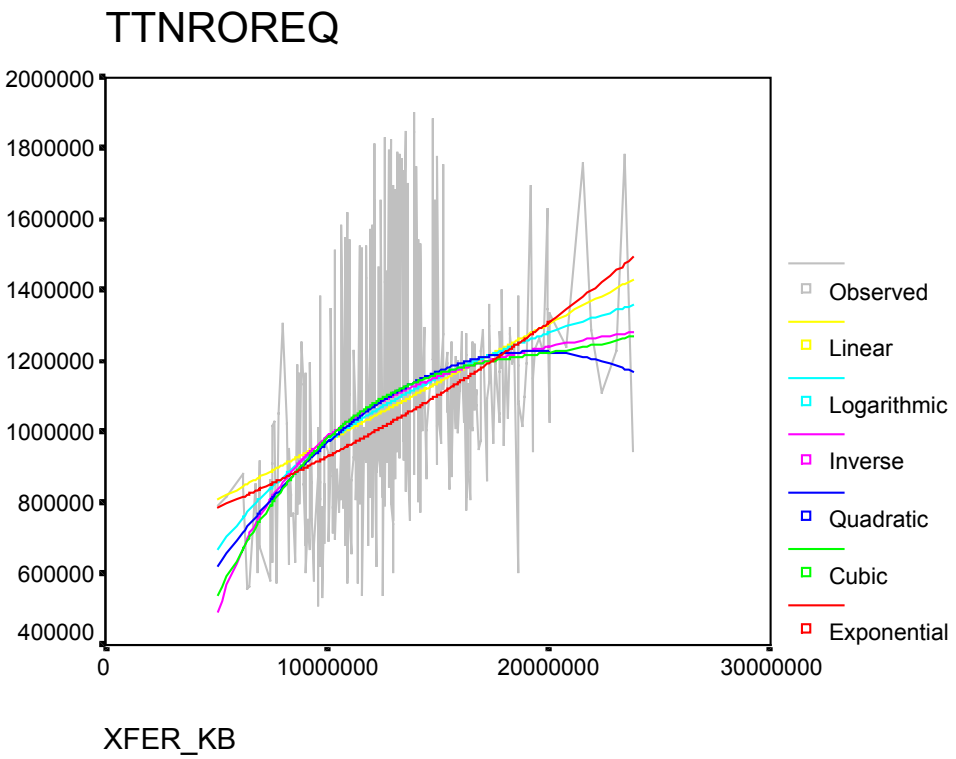

Figura 5.14 Regresión no lineal respecto a la transferencia total desde los servidores caché

Tabla 5.5 Parámetros obtenidos para la regresión no lineal de la figura 5.14

Método	Rsq	Pobl.	F	Sigf	b0	B1	b2	B2
LIN	.139	349	56.15	.000	642673	.0332		
LOG	.154	349	63.74	.000	-6.E+06	444031		
INV	.157	349	65.22	.000	1496663	-5.E+12		
CUA	.162	348	33.58	.000	122794	.1141	-3.E-09	
CUB	.164	347	22.72	.000	-307968	.2174	-1.E-08	1.8E-16
EXP	.175	349	74.29	.000	661733	3.4E-08		

El coeficiente de determinación Rsq se define como:

Variación explicada / Variación Total

Si ajustamos los datos referidos a Total de los Requerimientos efectuados a los servidores (TTNROREQ) como función de la transferencia total por cada uno de los servidores (XFER_KB), es de observarse que la función exponencial es la que muestra el mejor valor de Rsq, a pesar de ajustar moderadamente bien los datos.

También se ve que con un valor del estadístico de Fisher cercano a 75, valor más que aceptable para este estudio, y para n = 349 , alfa de 0,05 y grado de libertad (df) 2 , da 74,29 > 3.

Luego se puede rechazar la hipótesis nula de que la variable independiente seleccionada no mejora significativamente el modelo de regresión múltiple, por la hipótesis alternativa de que sí mejora.

Si ajustamos los datos referidos a Total Requerimiento (TTNROREQ) como función a Tráfico Directo desde los servidores WWW en internet, por haber obtenido un MISS en el Caché local (DFXFERKB) se observa que la función exponencial también ajusta bien los datos con un valor de Fisher cercano a 65, inferior al anterior análisis, pero aceptable a este estudio.

Tabla 5.6 Parámetros obtenidos para la regresión no lineal de la figura 5.15

Método	Rsq	Pobl.	F	Sigf	b0	B1	b2	B2
LIN	.125	349	49.71	.000	700528	.0337		
LOG	.146	349	59.82	.000	-5.E+06	400744		
INV	.156	349	64.68	.000	1471138	-4.E+12		
CUA	.158	348	32.72	.000	122794	.1311	-4.E-09	
CUB	.166	347	23,01	.000	160252	.3043	-2.E-08	3.9E-16
EXP	.158	349	65.33	.000	-473366	3.5E-08		

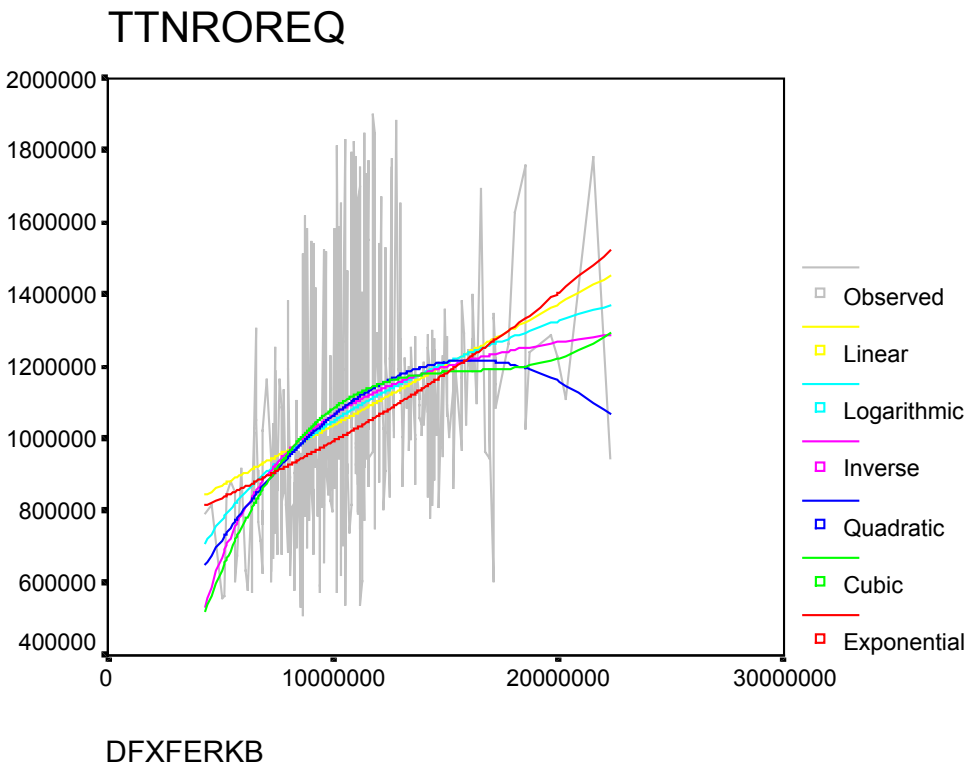

Figura 5.15 Regresión respecto a servidores originales

5.8 Resultados en la UNaM

En la Universidad Nacional de Misiones se han utilizado los resultados obtenidos por el servidor caché instalado en el Rectorado.

Si bien al momento de realizar estos estudios de los resultados se encuentra operativo un sistema cooperativo entre el Campus Universitario y la Facultad de Ciencias Exactas, Químicas y Naturales, donde se ha instalado un enlace a la red pública de “banda ancha” los resultados obtenidos hasta el momento no ameritan ser incluidos en este trabajo.

Habiendo considerado un período de 11 días entre el 18 y el 28 de Agosto, se

ha determinado que el servidor proxy de la Universidad de Misiones ha atendido 2403 requisitorias de servicio por hora, y ha efectuado transferencias horarias de 12,2 MB de Promedio.

Tabla 5.7 Resultados sobre caché de la UNaM

% de transferencia desde caché propio	*15,62*
Otros servidores	*0.28*
Total de Jerarquía	*15.9*

CAPÍTULO VI: Conclusiones y recomendaciones

6.1 Conclusiones

Con los resultados obtenidos podemos concluir en respuesta de los objetivos que cuando es aplicada, la cooperación de servidores Caché, esto es una mejora al rendimiento del sistema en su conjunto.

En el caso de la Universidad de Karlsruhe, la cooperación alcanzó valores entre 0,26 al 3,04% del total del tráfico distribuido. Ello implica que del tráfico solicitado a servidores de la Internet, aproximadamente 6 TB en el cuatrimestre en cuestión, el aporte de los servidores sibling llegaría a un máximo de 182 GB. Este número integra el 14,67% de eficiencia promedio de la jerarquía.

Cabe preguntarse en este punto, si es obtenido un beneficio del 12% en lo referente a ancho de banda ó montos a ser abonados a los distintos proveedores, porque no implementar un sistema de cooperación para poder obtener un 2% adicional, sin pérdida de rendimiento del sistema?.

Del ajuste de datos por regresión no lineal, se obtuvieron valores en un todo de acuerdo a Arlitt y Williamson [**30**] respecto a la distribución exponencial.

Es de esperarse que la conducta de los usuarios no varíe tanto con el correr del tiempo, razón por la cual no debería dedicarse la mayoría de los esfuerzos y recursos al diseño y evaluación de grandes esquemas cooperativos.

Los mayores beneficios son obtenidos por poblaciones relativamente pequeñas. Este resultado es confirmado con los valores obtenidos en la Universidad Nacional de Misiones donde se obtiene casi un 5% más de rendimiento para la misma herramienta.

Si bien a la Universidad de Karlsruhe, con la disponibilidad de ancho de banda que posee, le es indiferente la utilización o nó de herramientas como las propuestas en este trabajo, si obtuviésemos el mismo resultado de cooperación en la UNaM, el rendimiento del sistema para los usuarios rondaría un 20 %. Este “ancho de banda” adicional sería inmediatamente reconocida por los usuarios, como así

también las autoridades de la UNaM, quienes afrontan un elevado costo mensual para el mantenimiento del servicio.

6.2 Recomendaciones

Si bien se encuentra muy estudiado lo referente a cooperación de servidores caché, tanto en lo referente a simulaciones o instalaciones físicas; la mayoría de ellas se han realizado en instalaciones donde las limitaciones pasan a ser el rendimiento de los servidores y la memoria instalada, realidad muy alejada de instalaciones como las de la UNaM y otras instituciones como ella, donde el recurso más escaso es el ancho de banda. Por ello debería profundizarse el estudio de casos similares domo poder obtener resultados válido para la mayoría de Universidades en el marco de la Red de Interconexión Universitaria.

Si bien la cooperación es importante en la intranet, esta produce tráfico extra en esta. Es importante realizar estudios para profundizar este tema, para establecer en estos entornos que porcentaje de este tráfico es útil.

De la misma manera, hay que evaluar la instalación de tecnologías como ADSL, disponibles en casi todas las sedes de Universidades, como una alternativa de ampliación del ancho de banda y su implicancia con sistemas cooperativos de caché.

Se observan tanto en la Universidad de Karlsruhe como en la Universidad Nacional de Misiones, valores de indican la necesidad de desarrollar e implementar sistemas dinámicos que sean capaces de activar-desactivar los sistemas de cooperación.

BIBLIOGRAFÍA

(1) M. Baenstch, L. Baum, G. Molter, S. Rothkugel, P. Sturm. “World Wide Web Caching: The Application- Level View of the Internet”, IEEE 1997

(2) K. Bharat, A. Broder. “A Technique for Measuring the Relative Size and Overlap of Public Web Search Engines”. 7 International WWW Conference.

(3) A. Chankhunthod et al., "A Hierarchical Internet Object Cache" Usenix technical Conference”, San Diego 1997

(4) Wessels & Clafy , "Application of Internet Cache Protocol, version 2" Internet Engineerinf Task Force IETF.

(5) Squid Object Internet Cache. Online en http://ircache.nlanr.net

(6) Valloppillil & Rose . “Cache Array Routing Protocol”, Internet Draft

(7) Fan, Cao & Almeida, “Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol”

(8) Rousskov & Wessels, “Cache Digest“

(9) Makpamgou & Berengier, "Relais: Un protocole de maintien de cohérence de caches Web Coopérants"

(10) S. Gadde, M. Rabinovich, J. Chase. “Reduce, Reuse, Recycle: an approach to Building Large Internet Caches“

(11) National Laboratory for Applied Network Research (NLANR). Online en http://ircache.nlanr.net

(12) J. Gwertzman, M Seltzer , "The Case for Geographical push-caching". Workshop on hot operating Systems, 1995

(13) Zhang L., Floyd S., Jacobson V., "Adaptative Web Caching" 2nd Web Cache Workshop “ 1997

(14) RFC 2187, “Application of Internet Cache Protocol (ICP), version 2”

(15) Wessels & Clafy, “ICP and the Squid Web Cache, NSF”, National Science Foundation, 1997

(17) Krisnamurty, et al. “On the use and performance of content distribution networks“, ACM SIGCOMM, Internet Measurement Workshop, 2001

(18) Wolman Alec, Voelker Geoffrey, Sharma Nitin, Cardwell Neal, Karlin Anna and Levy Henry; “On the Scale and performance of cooperative proxy caching“, 17 ACM Symposium on Operating Systems Principles, 1999

(19) E-business Watch, “Bandwith Bloat”, march 12, 2001

(20) Gettys Jim, Nielsen Henrik, “The WebMUX protocol”, Internet Draft, 1999

(21) Allman Mark, Hayes Cris, Kruse Hans, Ostermann Shawn. "Tcp Performance over Satellite Links", First International Conference on Telecommunication Systems, 1997

(22) Girod Lewis, "Multiplexing of HTTP requests to improve throughput over high-delay links. Implemented as a modification of the Squid proxy cache", http://lecs.cs.ucla.edu/~girod/official/558-alt.ppt

(23) Postel J., “Internet Protocol”, RFC 791, ISI, University of Southern California, septiembre 1981.

(24) Postej J., “Transmission Control Protocol”, RFC 793, ISI, University of Southern California, septiembre 1981.

(25) Comer, Douglas S., “Internetworking with TCP/IP, Vol I, Principles, Protocols and Architecture”, 3rd. Edition

(26) Tannenbaum, Andew S., “Computer Networks”, Prentice Hall, 3rd. Edition, 1996

(27) Hunt, Craig, “TCP/IP Network Administration”, O’Reilly and Associates, Inc, 1994

(28) Stallings, William; “Comunicaciones y Redes de Computadores”, Prentice Hall, Quinta Edición, 1997

(29) Hunt, Craig, “Networkink Personal Computers with TCP/IP”, O’Reilly and Associates, Inc, 1995

(30) Almeida V, Bestavros A., Crovella M., de Oliveira A, “Characterizing Reference Locality in the WWW”, Proc. IEEE Conference on Parallel and Distributed System, 1996

(31) Arlitt M.F., Williamson C, “Web Server Workload Characterization: The Search for Invariants”, Proc. ACM SIGMETRICS, Philadelphia 1996.

(32) Bestavros A., “WWW Trafic Reduction, and Load Balancing through Server Based Caching”, IEEE Concurrency, 1997

(33) Squezer, paquete disponible desde http://www.uck.uni.torun.pl/~mac iek/w3cache/

ANEXO A

A.1 El mensaje ICP y sus códigos

El formato de los mensajes ICP están constituidos por una cabecera de 20 bytes fijos, a los cuales se adiciona un tamaño variable de añadido (payload).

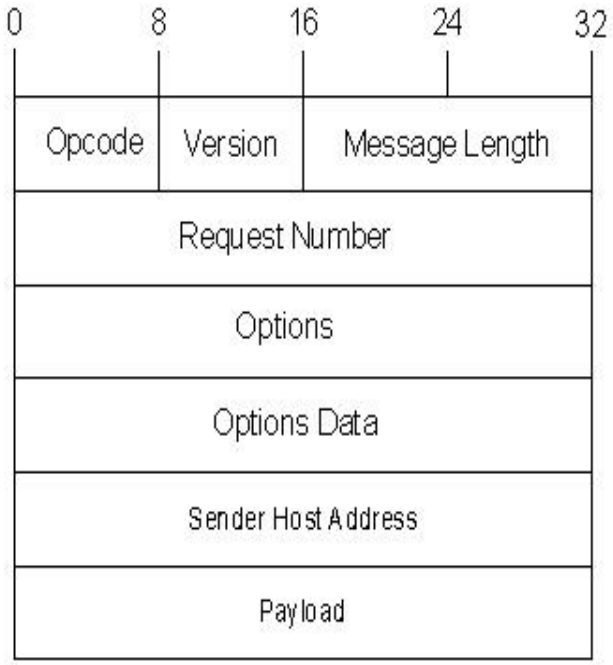

Figura A.1

Tabla A.1 Códigos de Operación ICP

OPCODE	*DEFINICION*
0	***ICP_OP_INVALID Mensaje inválido o completo con ceros.***
1	***ICP_OP_QUERY Mensaje de solicitud. En respuesta a este mensaje el servidor debe emitir un ICP_OP_HIT, ICP_OP_MISS, ICP_OP_ERR, ICP_OP_MISS_NOFETCH, ICP_OP_DENIED ó ICP_OP_HIT_OBJ.***
2	***ICP_OP_HIT El objeto se encuentra en el caché y puede ser obtenido por el solicitante.***
3	***ICP_OP_MISS El URL no existe en este Caché***
4	***ICP_OP_ERR Existe algún tipo de error en el procesamiento del mensaje, por ejemplo un URL inválido.***
5 a 9	***No Utilizado***
10	***ICP_OP_SECHO Es similar a ICP_OP_QUERY, pero para uso en simulaciones de solicitudes a servidores originales***
11	***ICP_OP_DECHO Es similar a ICP_OP_QUERY, utilizado en la simulación de solicitud a caches que no utilizan ICP.***
12 a 20	***No Utilizado***
21	***ICP_OP_MISS_NOFETCH indica que el caché está funcionando, pero en un estado donde no puede resolver el pedido. Por ejemplo en la fase de arranque cuando el cache esta reconstruyendo sus índices.***
22	***ICP_OP_DENIED Indica que el sitio solicitante no puede obtener el objeto del caché***
23	***ICP_OP_HIT_OBJ , similar a ICP_OP_HIT, pero los datos del objeto han sido incluídos en este mensaje de respuesta. Se utiliza cuando los objetos son lo suficientemente pequeños como para evitar la necesidad de transporte de otro mensaje.***

ANEXO B

B.1 Reporte de Squeezer para el Servidor Proxy1

B.1.1 Correspondiente al día 01 de Marzo de 2002

Tabla B.1 General information

Start date	Fri Mar 1 00:00:04 2002
End date	Sat Mar 2 00:00:02 2002
Total time	86397 s
HTTP requests per hour	48858
Transfer per hour	654635 kB

	No of req	Xfer (kB)	Xfer speed (kB/s)	Xfer %	Times to direct
Total traffic	1172585	15710871	5.465	100.00	**0.83**
Direct fetches	739908	13762084	6.604	87.60	1.00
This server	416829	1792546	2.785	11.41	0.42
Other servers	15848	154363	11.726	0.98	1.78
Cache hierarchy	432677	1946909	2.964	12.39	0.45

Tabla B.2 Sibling efficiency

Server	Current relation and options	HTTP	ICP	Xfer (KB)	Xfer %	No of req	Xfer speed (KB/s)	Times to direct
proxy4.rz.uni-karlsruhe.de	sibling	3128	3130	64116	0	4135	24.37	3.69
proxy2.rz.uni-karlsruhe.de	sibling	3128	3130	25818	0	4057	12.01	1.82
proxy3.rz.uni-karlsruhe.de	sibling	3128	3130	64428	0	5157	7.69	1.16
-		0	0	1877	0	2499	0.01	0.00

Tabla B.3 Refresh pattern efficiency

Refresh pattern	Options	No of req	Xfer (kB)	Req %	Xfer %	Xfer speed (kB/s)
^ftp:	1440 20% 10080	942	2545577	0.08	16.20	22.36
^gopher:	1440 0% 1440	0	0	0.00	0.00	0.00
.	0 30% 8640	1171643	13165294	99.92	83.80	4.77

Tabla B.4 By MIME type

MIME type	No of requests	Xfer (kB)	Requests %	Xfer %	Xfer speed (kB/s)
text/plain	14287	2522992	1.22	16.06	24.44
application/octet-stream	3826	1957298	0.33	12.46	7.06
text/html	264544	1832670	22.56	11.66	3.46
image/jpeg	159717	1795071	13.62	11.43	16.99
video/x-msvideo	1042	1666967	0.09	10.61	7.34
audio/mpeg	506	1201769	0.04	7.65	34.32
image/gif	496317	1168029	42.33	7.43	8.63
application/zip	267	855519	0.02	5.45	37.70
video/mpeg	1305	644545	0.11	4.10	38.16
-	176187	188306	15.03	1.20	0.15
application/pdf	1001	165419	0.09	1.05	42.91
video/quicktime	71	159665	0.01	1.02	119.98
application/x-debian-package	744	143140	0.06	0.91	11.99
application/x-zip-compressed	46	125407	0.00	0.80	3.99
application/x-tar	14	113036	0.00	0.72	27.65
audio/x-pn-realaudio	293	97939	0.02	0.62	31.37
application/x-shockwave-flash	3297	94824	0.28	0.60	33.36
video/msvideo	76	89905	0.01	0.57	44.31
application/x-download	18	88263	0.00	0.56	503.86
magnus-internal/cgi	2	84707	0.00	0.54	331.78
application/x-rpm	32	80095	0.00	0.51	261.60

audio/basic	104	72099	0.01	0.46	8.47
application/x-gzip	11	63213	0.00	0.40	178.27
application/x-javascript	25237	52365	2.15	0.33	4.03
multipart/byteranges	151	35727	0.01	0.23	46.04
text/css	11866	30885	1.01	0.20	7.11
image/pjpeg	615	30418	0.05	0.19	17.89
audio/x-mpeg	26	30399	0.00	0.19	17.96
multipart/x-byteranges	410	29651	0.03	0.19	19.64
video/x-ms-asf	178	29117	0.02	0.19	2.67
application/postscript	37	24241	0.00	0.15	189.58
video/x-ms-wmv	112	22249	0.01	0.14	52.39
application/vnd.ms-excel	14	22195	0.00	0.14	80.25
application/octetstream	51	21437	0.00	0.14	47.11
application/vnd.ms-powerpoint	30	21161	0.00	0.13	102.73
application/x-macbinary	1	21154	0.00	0.13	155.56
application/msword	67	18438	0.01	0.12	26.36
image/png	5341	16987	0.46	0.11	6.29
application/x-msdos-program	15	14531	0.00	0.09	19.11
application/realnetworksupgrade	16	13608	0.00	0.09	206.14
application/x-rar-compressed	15	12244	0.00	0.08	38.34
application/self-extracting	2	5571	0.00	0.04	36.58
application/excel	5	3941	0.00	0.03	90.87
image/tiff	2	3918	0.00	0.02	140.47

audio/midi	158	3253	0.01	0.02	55.28
application/x-compressed	34	3120	0.00	0.02	4.90
video/x-vcr	12	3069	0.00	0.02	29.75
text/xml	651	2475	0.06	0.02	6.49
image/bmp	75	1902	0.01	0.01	18.42
application/java-archive	10	1818	0.00	0.01	82.30
message/rfc822	29	1744	0.00	0.01	25.21
application/pkix-crl	8	1631	0.00	0.01	173.49
application/macbinary	1	1440	0.00	0.01	52.07
audio/x-wav	41	1262	0.00	0.01	4.11
/	15	1191	0.00	0.01	20.85
application/x-pointplus	317	1035	0.03	0.01	19.08
application/x-bzip2	1	959	0.00	0.01	16.84
application/mac-binhex40	1	877	0.00	0.01	67.04
application/x-msdownload	6	817	0.00	0.01	29.21
application/x-shswf	51	754	0.00	0.00	10.99
application/x-stuffit	2	746	0.00	0.00	79.83
audio/wav	24	742	0.00	0.00	108.02
application/x-java-archive	9	741	0.00	0.00	216.65
.swf	2	692	0.00	0.00	100.72
image/x-xbitmap	176	561	0.02	0.00	7.81
multipart/mixed	5	551	0.00	0.00	0.15
image/jpg	63	522	0.01	0.00	45.19

multipart/x-mixed-replace	3	472	0.00	0.00	0.20
application/x-chess-pgn	10	447	0.00	0.00	171.28
application/x-jar	4	421	0.00	0.00	137.52
text/javascript	220	420	0.02	0.00	3.26
gif89aÔ<	25	370	0.00	0.00	71.51
application/pps	1	367	0.00	0.00	221.31
application/x-director	7	361	0.00	0.00	101.11
aim/http	1104	359	0.09	0.00	0.37
application/x-msexcel	1	321	0.00	0.00	53.05
application/java-vm	40	298	0.00	0.00	59.32
application/x-java	43	246	0.00	0.00	12.06
audio/mid	30	231	0.00	0.00	11.64
text/x-component	29	225	0.00	0.00	56.42
application/ppt	1	223	0.00	0.00	5.90
application/x-webshots	4	202	0.00	0.00	32.96
img/*	9	179	0.00	0.00	22.49
x-text/plain	82	162	0.01	0.00	1.49
application/x-msn-messenger	555	157	0.05	0.00	1.42
application/java	52	148	0.00	0.00	31.01
image/x-ms-bmp	51	144	0.00	0.00	86.81
application/download	1	142	0.00	0.00	53.91
audio/x-ms-wma	4	133	0.00	0.00	111.55
application/x-sibelius-score	4	132	0.00	0.00	18.90

application/x-dvi	2	130	0.00	0.00	235.25
www/unknown	57	118	0.00	0.00	1.94
image/jpg.png.gif	21	108	0.00	0.00	111.01
application/x-rtf	1	96	0.00	0.00	19.07
gif	3	90	0.00	0.00	14.42
multipart/x-zip	5	81	0.00	0.00	6.40
application/x-tex	14	70	0.00	0.00	19.53
application/x-m3u	8	69	0.00	0.00	10.45
jpg	3	66	0.00	0.00	9.74
text/rtf	1	58	0.00	0.00	481.78
application/x-netgravity	110	47	0.01	0.00	1.63
video/x-ms-asx	61	45	0.01	0.00	1.78
none/none	1	45	0.00	0.00	0.01
application/downloadpdf	1	34	0.00	0.00	68.06
application/kdedownload	3	32	0.00	0.00	82.95
text/script	35	30	0.00	0.00	1.48
javascript/text	22	28	0.00	0.00	3.94
image/x-icon	11	27	0.00	0.00	10.13
application/vnd.rn-realplayer	25	26	0.00	0.00	0.27
image	38	26	0.00	0.00	14.56
video/mpg	1	25	0.00	0.00	6.08
application/smil	25	25	0.00	0.00	2.44
text/axp	4	24	0.00	0.00	16.28

application/x-gzip-compressed	1	24	0.00	0.00	99.18
image/tif	1	21	0.00	0.00	431.22
app/x-hotbar-xip20	18	20	0.00	0.00	4.08
java/*	3	17	0.00	0.00	1.42
image/	30	17	0.00	0.00	1.32
application/force-download	1	15	0.00	0.00	29.27
image/x-portable-graymap	19	14	0.00	0.00	90.43
jrun-internal/target	3	13	0.00	0.00	223.63
application/font-tdpfr	1	12	0.00	0.00	389.59
pdf	1	11	0.00	0.00	15.33
application/vnd.mozilla.xul+xml	3	11	0.00	0.00	222.68
magti_banner.gif	1	11	0.00	0.00	0.80
tv_banner.gif	1	10	0.00	0.00	0.94
audio/x-pn-realaudio-plugin	25	10	0.00	0.00	0.51
kedi_banner.gif	1	9	0.00	0.00	0.61
application/x-java-applet	1	9	0.00	0.00	262.10
application/x-perl	22	8	0.00	0.00	0.40
images/gif	4	7	0.00	0.00	17.18
text/text	15	6	0.00	0.00	1.72
unknown	9	6	0.00	0.00	3.50
application/x-shockwave-flash2-preview	1	6	0.00	0.00	123.12
image/x-png	3	5	0.00	0.00	6.80
text/rfc822	2	4	0.00	0.00	10.28

audio/x-scpls	14	4	0.00	0.00	0.24
application/unknown	1	4	0.00	0.00	56.27
application/image	2	4	0.00	0.00	8.25
image/*	2	3	0.00	0.00	11.83
application/x-java-vm	1	3	0.00	0.00	76.17
application/x-wais-source	4	3	0.00	0.00	28.24
application/x-pkcs7-signature	1	3	0.00	0.00	6.82
application/vnd.rn-rn_music_package	2	3	0.00	0.00	3.24
audio/x-midi	1	2	0.00	0.00	8.33
text/x-javascript	5	2	0.00	0.00	0.18
video/x-ms-wvx	4	2	0.00	0.00	2.97
text/http	8	2	0.00	0.00	1.39
application/x-httpd-php	2	2	0.00	0.00	23.92
magnus-internal/cold-fusion	4	1	0.00	0.00	0.77
text/x-vcard	3	1	0.00	0.00	2.00
caramail/download	4	1	0.00	0.00	0.03
application/x-icq	4	1	0.00	0.00	0.40
application/x-java-jnlp-file	5	1	0.00	0.00	1.13
cnn/user-cookie	4	1	0.00	0.00	2.02
application/vndms-pkiseccat	4	1	0.00	0.00	28.65
image/vnd.wap.wbmp	4	1	0.00	0.00	7.07
video/s-ms-asf	2	1	0.00	0.00	7.03
text/txt	2	1	0.00	0.00	0.14

application/javascript	1	0	0.00	0.00	2.29
application/x-httpd-cgi	2	0	0.00	0.00	0.26
application/x-quicktime-response	3	0	0.00	0.00	0.56
application/x-www-form-urlencoded	1	0	0.00	0.00	1.64
message/delivery-status	1	0	0.00	0.00	1.24
audio/x-mpegurl	2	0	0.00	0.00	0.03
audio/x-realaudio	1	0	0.00	0.00	1.01
text/x-unknown-content-type	1	0	0.00	0.00	0.92
text/cgi	1	0	0.00	0.00	0.16
text/devil	1	0	0.00	0.00	0.49
text/js	1	0	0.00	0.00	13.67

Tabla B.5 By cache result codes

Cache result code	No of req	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TCP_MEM_HIT	37378	3.19	68056	0.43	170.21
TCP_IMS_HIT	154341	13.16	39135	0.25	17.93
TCP_REFRESH_HIT	44574	3.80	418449	2.66	5.83
TCP_REFRESH_MISS	16316	1.39	118106	0.75	6.48
TCP_DENIED	852	0.07	911	0.01	248.13
TCP_CLIENT_REFRESH_MISS	104032	8.87	146708	0.93	2.43
TCP_HIT	200044	17.06	1663427	10.59	33.23
TCP_NEGATIVE_HIT	4942	0.42	6821	0.04	137.61
TCP_MISS	610106	52.03	13249254	84.33	4.96

Tabla B.6 By peer status

Peer status	Req by fetch	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TIMEOUT_NONE	2499	0.21	1877	0.01	0.01
DIRECT	574514	49.00	8558422	54.47	4.87
TIMEOUT_DIRECT	165394	14.11	5203661	33.12	15.96
SIBLING_HIT	13349	1.14	154363	0.98	11.73
NONE	416829	35.55	1792546	11.41	2.79

Tabla B.7 Performance

User time	130.08 s
System time	23.61 s
Children time	0.00 s
Children system time	0.00 s
Together	**153.69** s
Performance	**7629 lines/s**

running on **Linux proxy1 2.4.18 #1 SMP Thu Feb 28 10:53:37 CET 2002 i686 unknown**

---

squeezer.pl v. 0.31 M.K., last modified: Mon Aug 16 13:00:21 1999

Tutorial on using squeezer generated data

ANEXO C

C.1 Reporte de Squeezer para el Servidor Proxy2

C.1.1 Correspondiente al día 01 de Marzo de 2002

Tabla C.1 General information

Start date	Fri Mar 1 00:00:15 2002
End date	Sat Mar 2 00:00:01 2002
Total time	86386 s
HTTP requests per hour	68884
Transfer per hour	519085 kB

	No of req	Xfer (kB)	Xfer speed (kB/s)	Xfer %	Times to direct
Total traffic	1652952	12456044	8.623	100.00	**0.97**
Direct fetches	983944	10799862	8.884	86.70	1.00
This server	660888	1570331	8.494	12.61	0.96
Other servers	8120	84734	20.445	0.68	2.30
Cache hierarchy	669008	1655066	8.756	13.29	0.99

Tabla C.2 Sibling efficiency

Server	Current relation and options	HTTP	ICP	Xfer (KB)	Xfer %	No of req	Xfer speed (KB/s)	Times to direct
proxy3.rz.uni-karlsruhe.de	sibling	3128	3130	29201	0	2395	101.35	11.41
proxy4.rz.uni-karlsruhe.de	sibling	3128	3130	15902	0	2567	72.99	8.22
proxy1.rz.uni-karlsruhe.de		0	0	39630	0	1688	10.89	1.23
-		0	0	1116	0	1470	0.03	0.00

Tabla C.3 Refresh pattern efficiency

Refresh pattern	Options	No of req	Xfer (kB)	Req %	Xfer %	Xfer speed (kB/s)
^ftp:	1440 20% 10080	843	2455973	0.05	19.72	36.76
^gopher:	1440 0% 1440	1	47	0.00	0.00	30.52
.	0 30% 8640	1652108	10000024	99.95	80.28	7.26

Tabla C.4 By MIME type

MIME type	No of requests	Xfer (kB)	Requests %	Xfer %	Xfer speed (kB/s)
text/html	364091	3300959	22.03	26.50	7.91
application/octet-stream	4059	1603462	0.25	12.87	21.87
image/jpeg	182994	1461095	11.07	11.73	21.61
image/gif	802412	1448676	48.54	11.63	17.82

text/plain	39761	1288830	2.41	10.35	28.10
-	169695	633821	10.27	5.09	1.04
video/x-msvideo	93	602024	0.01	4.83	21.25
application/zip	336	402743	0.02	3.23	28.26
application/download	51	270913	0.00	2.17	554.66
video/mpeg	468	251787	0.03	2.02	47.97
application/x-zip-compressed	63	187568	0.00	1.51	89.38
audio/mpeg	84	142912	0.01	1.15	13.78
application/pdf	996	118765	0.06	0.95	36.81
application/x-shockwave-flash	4081	98051	0.25	0.79	22.59
video/quicktime	76	97735	0.00	0.78	104.77
application/x-javascript	48475	92666	2.93	0.74	11.04
application/x-tar	28	63348	0.00	0.51	73.51
audio/x-pn-realaudio	192	51126	0.01	0.41	32.30
text/css	22792	43209	1.38	0.35	24.82
multipart/byteranges	430	35600	0.03	0.29	109.79
audio/x-mpeg	22	23529	0.00	0.19	1.74
multipart/x-byteranges	197	19084	0.01	0.15	3.12
image/png	4764	18138	0.29	0.15	11.74
video/x-ms-wmv	18	16265	0.00	0.13	398.46
application/postscript	64	16222	0.00	0.13	27.62
application/x-msdos-program	22	13915	0.00	0.11	69.66
application/msword	138	13552	0.01	0.11	18.01

image/pjpeg	636	12910	0.04	0.10	26.41
application/unix-tar	9	12875	0.00	0.10	80.96
image/tiff	42	12226	0.00	0.10	158.30
image/bmp	137	12203	0.01	0.10	149.37
application/vnd.ms-powerpoint	16	11754	0.00	0.09	52.70
application/x-rpm	3	9080	0.00	0.07	403.52
application/vnd.ms-excel	304	8006	0.02	0.06	120.06
video/msvideo	2	5428	0.00	0.04	92.64
application/x-gzip	6	5108	0.00	0.04	170.85
application/x-download	1	5014	0.00	0.04	107.06
audio/midi	185	4497	0.01	0.04	32.70
application/pkix-crl	22	4255	0.00	0.03	38.68
image/jpg	294	3749	0.02	0.03	43.21
application/self-extracting	1	3413	0.00	0.03	305.42
application/x-debian-package	19	3035	0.00	0.02	37.05
application/x-rar-compressed	5	2619	0.00	0.02	15.26
video/x-ms-asf	31	2590	0.00	0.02	9.34
text/xml	1136	2551	0.07	0.02	5.09
audio/x-ms-wma	2	2118	0.00	0.02	1409.58
audio/x-wav	69	2048	0.00	0.02	26.04
text/javascript	941	1481	0.06	0.01	18.43
video/x-vcr	6	1099	0.00	0.01	31.77
application/x-java-archive	4	1053	0.00	0.01	68.83

multipart/mixed	10	1033	0.00	0.01	0.03
audio/basic	109	816	0.01	0.01	41.81
application/vnd.rn-realmedia	2	760	0.00	0.01	4.27
application/x-msword	3	622	0.00	0.00	255.24
application/vnd.ms-asf	1	558	0.00	0.00	80.20
application/realnetworksupgrade	4	552	0.00	0.00	61.61
application/x-bzip2	1	502	0.00	0.00	10.26
application/x-pointplus	201	491	0.01	0.00	3.49
www/unknown	77	473	0.00	0.00	42.58
application/x-zip	22	407	0.00	0.00	19.09
application/jar	5	337	0.00	0.00	273.10
aim/http	807	316	0.05	0.00	0.26
image/jpg.png.gif	56	293	0.00	0.00	63.37
message/rfc822	8	258	0.00	0.00	30.75
application/x-compressed	50	226	0.00	0.00	3.44
application/x-chess-pgn	3	186	0.00	0.00	357.24
application/x-ipix	1	180	0.00	0.00	506.79
application/x-netgravity	373	166	0.02	0.00	3.84
application/x-shockwave-flash2-preview	4	163	0.00	0.00	6.34
application/x-java	24	138	0.00	0.00	56.78
application/x-shswf	6	130	0.00	0.00	29.95
images/gif	60	121	0.00	0.00	18.84
application/x-webshots	2	116	0.00	0.00	58.04

application/octet-string	1	110	0.00	0.00	692.40
application/rtf	5	104	0.00	0.00	20.44
img/*	5	104	0.00	0.00	5.93
jpg	2	94	0.00	0.00	98.29
application/octetstream	10	92	0.00	0.00	47.38
/	3	87	0.00	0.00	35.28
application/java-archive	3	86	0.00	0.00	73.37
.swf	1	77	0.00	0.00	69.59
text/rfc822	2	76	0.00	0.00	128.47
application/java-vm	22	66	0.00	0.00	35.66
application/x-gzip-compressed	3	65	0.00	0.00	215.65
application/pdb	1	51	0.00	0.00	41.86
application/x-director	3	50	0.00	0.00	163.95
application/x-msexcel	2	49	0.00	0.00	45.49
audio/x-pn-realaudio-plugin	39	48	0.00	0.00	3.66
audio/mid	4	48	0.00	0.00	136.03
application/x-msn-messenger	140	43	0.01	0.00	1.02
text/*	2	42	0.00	0.00	5.53
image/x-xbitmap	65	42	0.00	0.00	2.33
application/x-java-vm	3	38	0.00	0.00	42.09
application-x/javascript	62	38	0.00	0.00	2.26
text/x-component	5	35	0.00	0.00	23.58
application/powerpoint	1	35	0.00	0.00	3.44

text/http	28	34	0.00	0.00	7.60
application/x-macbinary	1	31	0.00	0.00	6.62
image/x-icon	18	29	0.00	0.00	20.40
image	77	27	0.00	0.00	0.64
audio/x-midi	2	26	0.00	0.00	1.11
javascript/text	20	25	0.00	0.00	3.20
application/x-m3u	4	24	0.00	0.00	6.79
text/axp	4	24	0.00	0.00	16.00
multipart/x-mixed-replace	2	22	0.00	0.00	0.05
x-text/plain	4	20	0.00	0.00	7.25
application/vnd.rn-realplayer	28	20	0.00	0.00	0.81
application/x-perl	43	18	0.00	0.00	0.79
app/x-hotbar-xip20	21	18	0.00	0.00	2.03
application/x-jar	1	17	0.00	0.00	262.78
application/java	25	16	0.00	0.00	6.25
image/x-ms-bmp	1	16	0.00	0.00	273.32
application/x-vermeer-rpc	31	15	0.00	0.00	0.71
application/applefile	1	14	0.00	0.00	7.32
pdf	1	11	0.00	0.00	35.89
application/futuresplash	1	9	0.00	0.00	130.61
image/x-guffaw	1	9	0.00	0.00	14.61
image/	16	9	0.00	0.00	1.37
application/x-futuresplash	1	8	0.00	0.00	26.66

application/x-dvi	1	7	0.00	0.00	2.30
application/x-x509-ca-cert	5	7	0.00	0.00	2.32
video/x-ms-wvx	15	7	0.00	0.00	12.64
audio/x-aiff	1	7	0.00	0.00	238.12
unknown	8	5	0.00	0.00	2.59
:	15	5	0.00	0.00	6.54
audio/x-realaudio	2	5	0.00	0.00	17.33
gif	8	5	0.00	0.00	2.37
application/smil	5	4	0.00	0.00	2.99
image/x-portable-graymap	1	4	0.00	0.00	67.61
video/x-ms-asx	6	4	0.00	0.00	2.34
text/vnd.wap.wml	2	4	0.00	0.00	3.55
application/vnd.mozilla.xul+xml	1	3	0.00	0.00	11.27
application/gzip	1	2	0.00	0.00	8.55
img/png	1	2	0.00	0.00	0.81
application/x-www-form-urlencoded	6	2	0.00	0.00	1.02
video/avi	1	2	0.00	0.00	0.07
application/x-icq	6	2	0.00	0.00	1.18
text/x-c	2	2	0.00	0.00	0.94
text/text	1	1	0.00	0.00	4.13
cnn/user-cookie	6	1	0.00	0.00	1.17
text/x-unknown-content-type	6	1	0.00	0.00	0.30
magnus-internal/cold-fusion	6	1	0.00	0.00	1.80

application/x-httpd-cgi	6	1	0.00	0.00	0.89
application/x-quicktime-response	2	1	0.00	0.00	1.63
application/ms-tnef	1	1	0.00	0.00	0.61
application/x-ns-proxy-autoconfig	1	1	0.00	0.00	151.00
text/devil	2	0	0.00	0.00	0.01
image/xbm	1	0	0.00	0.00	1.19
magnus-internal/directory	3	0	0.00	0.00	4.70
img/gif	2	0	0.00	0.00	3.80
application/vnd.netfpx	2	0	0.00	0.00	1.81
application/x-msdownload	1	0	0.00	0.00	8.83
magnus-internal/parsed-html	1	0	0.00	0.00	1.70
x-application/cnf	1	0	0.00	0.00	13.32
application/eiskonzept	1	0	0.00	0.00	3.02
audio/x-mpegurl	1	0	0.00	0.00	3.26
application/x-java-jnlp-file	1	0	0.00	0.00	0.88
application/gif	1	0	0.00	0.00	3.01
java/*	1	0	0.00	0.00	14.59
application/vndms-pkiseccat	1	0	0.00	0.00	26.26

Tabla C.5 By cache result codes

Cache result code	No of req	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TCP_MEM_HIT	79360	4.80	166937	1.34	157.25
TCP_IMS_HIT	296352	17.93	87787	0.70	14.22
TCP_REFRESH_HIT	70288	4.25	369802	2.97	9.59
TCP_REFRESH_MISS	24650	1.49	269695	2.17	20.63
TCP_DENIED	427	0.03	456	0.00	117.54
TCP_CLIENT_REFRESH_MISS	136769	8.27	233162	1.87	8.30
TCP_HIT	268293	16.23	1292603	10.38	42.75
TCP_NEGATIVE_HIT	4973	0.30	9444	0.08	159.36
TCP_MISS	771825	46.69	10022541	80.46	7.55
TCP_SWAPFAIL_MISS	15	0.00	3613	0.03	163.52

Tabla C.6 By peer status

Peer status	Req by fetch	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TIMEOUT_NONE	1470	0.09	1116	0.01	0.03
DIRECT	761606	46.08	7085655	56.89	6.85
TIMEOUT_DIRECT	222338	13.45	3714207	29.82	20.57
SIBLING_HIT	6650	0.40	84734	0.68	20.45
NONE	660888	39.98	1570331	12.61	8.49

Tabla C.7 Performance

User time	202.30 s
System time	7.00 s
Children time	0.00 s
Children system time	0.00 s
Together	**209.30** s
Performance	**7897 lines/s**

running on **Linux proxy2 2.2.17 #1 Sun Jun 25 09:24:41 EST 2000 i686 unknown**

---

squeezer.pl v. 0.31 M.K., last modified: Mon Aug 16 13:00:21 1999

Tutorial on using squeezer generated data

ANEXO D

D.1 Reporte de Squeezer para el Servidor Proxy3

D.1.1 Correspondiente al día 01 de Marzo de 2002

Tabla D.1 General information

Start date	Fri Mar 1 00:00:09 2002
End date	Sat Mar 2 00:00:02 2002
Total time	86392 s
HTTP requests per hour	32957
Transfer per hour	428650 kB

	No of req	Xfer (kB)	Xfer speed (kB/s)	Xfer %	Times to direct
Total traffic	790923	10286759	7.229	100.00	**0.98**
Direct fetches	500942	8759917	7.358	85.16	1.00
This server	278601	1411143	7.727	13.72	1.05
Other servers	11380	114741	27.343	1.12	3.72
Cache hierarchy	289981	1525884	8.168	14.83	1.11

Tabla D.2 Sibling efficiency

Server	Current relation and options	HTTP	ICP	Xfer (KB)	Xfer %	No of req	Xfer speed (KB/s)	Times to direct
proxy4.rz.uni-karlsruhe.de	sibling	3128	3130	86582	0	5901	40.78	5.54
proxy2.rz.uni-karlsruhe.de	sibling	3128	3130	11764	0	1998	18.03	2.45
proxy1.rz.uni-karlsruhe.de		0	0	16393	0	1821	11.54	1.57
-		0	0	957	0	1660	0.02	0.00

Tabla D.3 Refresh pattern efficiency

Refresh pattern	Options	No of req	Xfer (kB)	Req %	Xfer %	Xfer speed (kB/s)
^ftp:	1440 20% 10080	347	1250423	0.04	12.16	24.16
^gopher:	1440 0% 1440	0	0	0.00	0.00	0.00
.	0 30% 8640	790576	9036335	99.96	87.84	6.59

Tabla D.4 By MIME type:

MIME type	No of requests	Xfer (kB)	Requests %	Xfer %	Xfer speed (kB/s)
application/octet-stream	6079	3160840	0.77	30.73	13.51
image/jpeg	143034	1310450	18.08	12.74	9.02
text/html	174975	1309171	22.12	12.73	4.74
text/plain	10887	868233	1.38	8.44	16.44

image/gif	311860	732527	39.43	7.12	7.90
audio/midi	301	539198	0.04	5.24	5.93
video/mpeg	974	480209	0.12	4.67	28.19
video/x-msvideo	232	467094	0.03	4.54	36.80
application/x-zip-compressed	42	228215	0.01	2.22	32.06
audio/mpeg	84	168022	0.01	1.63	16.45
application/x-mpeg	55	132834	0.01	1.29	14.45
application/zip	193	127030	0.02	1.23	19.63
application/x-shockwave-flash	2963	93031	0.37	0.90	32.23
application/downloadable	15	87191	0.00	0.85	60.25
audio/x-pn-realaudio	204	86075	0.03	0.84	17.31
-	92988	80970	11.76	0.79	0.19
application/x-javascript	28689	72473	3.63	0.70	7.41
video/quicktime	33	52007	0.00	0.51	102.40
video/msvideo	4	50260	0.00	0.49	90.69
application/internet-property-stream	239	46590	0.03	0.45	27.19
application/pdf	169	27192	0.02	0.26	14.93
text/css	8055	19453	1.02	0.19	9.32
application/x-macbinary	1	15005	0.00	0.15	78.39
video/x-ms-wmv	16	12269	0.00	0.12	162.72
audio/x-mpeg	7	11810	0.00	0.11	65.34
application/x-msdos-program	8	11180	0.00	0.11	2.48
application/x-tar	6	10365	0.00	0.10	62.61

text/xml	2807	10138	0.35	0.10	6.90
multipart/byteranges	64	8531	0.01	0.08	64.11
video/x-ms-asf	234	8134	0.03	0.08	3.32
application/x-debian-package	39	5638	0.00	0.05	52.46
image/png	2128	5144	0.27	0.05	3.63
application/x-compress	1	4586	0.00	0.04	61.67
image/pjpeg	117	3486	0.01	0.03	28.31
image/bmp	112	3373	0.01	0.03	24.62
application/jar	20	2941	0.00	0.03	471.48
application/msword	21	2466	0.00	0.02	53.81
application/x-gzip	11	2313	0.00	0.02	85.37
application/mac-binhex40	2	2251	0.00	0.02	1.91
audio/x-ms-wma	7	2039	0.00	0.02	2.72
multipart/x-byteranges	81	1816	0.01	0.02	41.45
application/binary	1	1684	0.00	0.02	314.48
audio/basic	230	1420	0.03	0.01	32.50
application/realnetworksupgrade	12	1346	0.00	0.01	61.15
application/x-compressed	100	1055	0.01	0.01	14.16
application/x-director	11	917	0.00	0.01	63.10
image/jpg	63	856	0.01	0.01	4.98
audio/x-wav	29	810	0.00	0.01	40.61
application/vnd.ms-powerpoint	5	782	0.00	0.01	39.78
multipart/x-mixed-replace	6	762	0.00	0.01	10.58

application/postscript	4	751	0.00	0.01	0.80
application/x-stuffit	1	743	0.00	0.01	21.30
image/tiff	8	737	0.00	0.01	54.10
interface/x-winamp-skin	6	697	0.00	0.01	107.01
message/rfc822	59	664	0.01	0.01	6.53
application/vnd.rn-realmedia	15	659	0.00	0.01	0.42
application/futuresplash	8	650	0.00	0.01	377.08
image/*	31	619	0.00	0.01	14.85
text/x-component	130	612	0.02	0.01	230.21
application/cab	6	576	0.00	0.01	347.02
video/x-vcr	2	575	0.00	0.01	23.73
application/stuffit	1	490	0.00	0.00	36.45
aim/http	1056	420	0.13	0.00	0.25
application/x-rar-compressed	1	377	0.00	0.00	6.96
application/mixed	3	371	0.00	0.00	16.55
application/macbinary	1	369	0.00	0.00	1.90
application/x-pointplus	137	366	0.02	0.00	17.12
application/x-java-archive	6	347	0.00	0.00	25.99
application/wbload	2	302	0.00	0.00	73.03
application/x-java-class	47	252	0.01	0.00	10.58
application/x-webshots	4	241	0.00	0.00	21.07
audio/x-midi	25	239	0.00	0.00	21.69
application/unknown	2	233	0.00	0.00	30.84

application/vnd.ms-excel	3	194	0.00	0.00	18.35
application/java-vm	56	177	0.01	0.00	22.74
application/vnd.ms-access	1	162	0.00	0.00	8.59
img/*	9	160	0.00	0.00	11.57
text/javascript	105	148	0.01	0.00	5.15
application/rtf	3	146	0.00	0.00	39.67
application/octetstream	7	145	0.00	0.00	51.32
image/x-bitmap	1	133	0.00	0.00	102.13
image/jpg-gif	2	130	0.00	0.00	2.42
audio/mid	13	129	0.00	0.00	2.03
application/x-netgravity	234	119	0.03	0.00	3.53
application/x-shswf	4	107	0.00	0.00	17.79
application/x-java	11	102	0.00	0.00	93.42
application/x-mss-scancfg	1	95	0.00	0.00	30.62
image/tif	1	90	0.00	0.00	232.78
application/x-bzip2	2	81	0.00	0.00	2.09
application/download	1	69	0.00	0.00	145.11
image	38	66	0.00	0.00	25.91
application/force-download	3	64	0.00	0.00	17.90
application/x-zip	2	56	0.00	0.00	146.85
www/unknown	28	53	0.00	0.00	9.64
application/font-tdpfr	6	50	0.00	0.00	182.86
app/x-hotbar-xip20	90	48	0.01	0.00	2.23

unknown	52	40	0.01	0.00	1.18
text/axp	6	37	0.00	0.00	16.37
/	2	36	0.00	0.00	47.50
application/java	4	34	0.00	0.00	40.94
application/pkix-crl	16	28	0.00	0.00	4.92
image/x-xbitmap	48	27	0.01	0.00	1.12
application/x-jar	1	23	0.00	0.00	68.91
application/x-m3u	3	23	0.00	0.00	9.55
audio/wav	4	23	0.00	0.00	124.76
text/http	19	22	0.00	0.00	14.72
audio/x-pn-realaudio-plugin	57	21	0.01	0.00	0.41
application/kdedownload	2	21	0.00	0.00	108.08
application/x-msdownload	1	20	0.00	0.00	17.62
application/java-archive	5	20	0.00	0.00	29.76
text/rtf	1	20	0.00	0.00	15.61
application/x-msn-messenger	66	19	0.01	0.00	0.99
java/*	5	17	0.00	0.00	28.97
image/x-icon	9	16	0.00	0.00	10.96
application/x-mobipocket-subscription	7	16	0.00	0.00	32.14
images/gif	8	16	0.00	0.00	8.78
application/msexcel	1	15	0.00	0.00	7.74
image/vnd.wap.wbmp	20	13	0.00	0.00	0.11
image/swf	5	13	0.00	0.00	0.45

application/vnd.rn-realplayer	13	13	0.00	0.00	0.77
text/x-csv	2	12	0.00	0.00	10.90
application/smil	11	11	0.00	0.00	9.55
text/js	10	11	0.00	0.00	7.87
text/text	7	10	0.00	0.00	3.16
x-world/x-vrml	2	8	0.00	0.00	28.91
application/javascript	4	8	0.00	0.00	0.99
application/vnd.ms-asf	1	7	0.00	0.00	6.73
application/x-palm-dba	1	7	0.00	0.00	13.09
image/x-guffaw	1	7	0.00	0.00	22.43
application/x-httpd-cgi	15	6	0.00	0.00	1.23
image/	11	6	0.00	0.00	1.07
text	7	5	0.00	0.00	5.09
text/devil	17	5	0.00	0.00	0.91
text/x-ldif	1	4	0.00	0.00	8.74
application/vnd.rn-rn_music_package	3	4	0.00	0.00	4.26
application/x-java-jnlp-file	4	3	0.00	0.00	3.65
application/x-shockwave-flash2-preview	1	3	0.00	0.00	225.59
binary/octet-stream	1	3	0.00	0.00	7.37
cnn/user-cookie	11	3	0.00	0.00	1.32
text/*	1	3	0.00	0.00	10.07
application/x-perl	9	3	0.00	0.00	1.71
application/x-chess-pgn	1	3	0.00	0.00	51.81

application/x-unknown	2	2	0.00	0.00	0.12
application/vndms-pkiseccat	8	2	0.00	0.00	9.24
application/x-httpd-php	2	2	0.00	0.00	30.08
application/x-quicktime-response	2	2	0.00	0.00	1.90
application/x-www-form-urlencoded	3	2	0.00	0.00	3.16
application/x-vermeer-rpc	6	2	0.00	0.00	5.60
application/x-httpd-php3	1	2	0.00	0.00	3.35
application/x-dvi	1	2	0.00	0.00	2.60
img/gif	7	1	0.00	0.00	3.73
application/x-mplayer2	2	1	0.00	0.00	0.76
application/x-wais-source	2	1	0.00	0.00	28.52
application/x-icq	4	1	0.00	0.00	1.63
application/x-troff	2	1	0.00	0.00	3.05
magnus-internal/notfemail	3	1	0.00	0.00	0.22
image/xbm	1	0	0.00	0.00	2.35
:	2	0	0.00	0.00	5.09
7am/clockfeed	4	0	0.00	0.00	0.52
httpd/unix-directory	2	0	0.00	0.00	0.39
magnus-internal/cold-fusion	2	0	0.00	0.00	0.34
x-application/cnf	1	0	0.00	0.00	7.31

Tabla D.5 By cache result codes

Cache result code	No of req	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TCP_MEM_HIT	16723	2.11	56699	0.55	244.71
TCP_REFRESH_HIT	48957	6.19	279314	2.72	7.31
TCP_IMS_HIT	109284	13.82	29909	0.29	17.48
TCP_REFRESH_MISS	16264	2.06	97160	0.94	10.94
TCP_DENIED	64	0.01	70	0.00	266.24
TCP_CLIENT_REFRESH_MISS	37467	4.74	118632	1.15	5.12
TCP_HIT	132373	16.74	1304751	12.68	63.27
TCP_NEGATIVE_HIT	6371	0.81	4805	0.05	253.49
TCP_MISS	423417	53.53	8395409	81.61	6.31
TCP_SWAPFAIL_MISS	3	0.00	4	0.00	29.87

Tabla D.6 By peer status

Peer status	Req by fetch	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TIMEOUT_NONE	1660	0.21	957	0.01	0.02
DIRECT	354458	44.82	6239018	60.65	6.28
TIMEOUT_DIRECT	146484	18.52	2520898	24.51	12.75
SIBLING_HIT	9720	1.23	114741	1.12	27.34
NONE	278601	35.22	1411143	13.72	7.73

Tabla D.7 Performance

User time	90.22 s
System time	12.19 s
Children time	0.00 s
Children system time	0.00 s
Together	**102.41** s
Performance	**7723 lines/s**

running on **Linux proxy3 2.2.17 #1 Sun Jun 25 09:24:41 EST 2000 i686 unknown**

---

squeezer.pl v. 0.31 M.K., last modified: Mon Aug 16 13:00:21 1999

Tutorial on using squeezer generated data

ANEXO E

E.1 Reporte de Squeezer para el Servidor Proxy4

E.1.1 Correspondiente al día 01 de Marzo de 2002

Tabla E.1 General information

Start date	Fri Mar 1 00:00:10 2002
End date	Sat Mar 2 00:00:01 2002
Total time	86391 s
HTTP requests per hour	45600
Transfer per hour	657972 kB

	No of req	Xfer (kB)	Xfer speed (kB/s)	Xfer %	Times to direct
Total traffic	1094313	15789771	7.229	100.00	**0.94**
Direct fetches	790672	13814468	7.706	87.49	1.00
This server	281332	1764232	5.516	11.17	0.72
Other servers	22309	209658	26.093	1.33	3.39
Cache hierarchy	303641	1973891	6.020	12.50	0.78

Tabla E.2 Sibling efficiency

Server	Current relation and options	HTTP	ICP	Xfer (KB)	Xfer %	No of req	Xfer speed (KB/s)	Times to direct
proxy1.rz.uni-karlsruhe.de		0	0	63702	0	4159	27.70	3.59
proxy2.rz.uni-karlsruhe.de	sibling	3128	3130	52496	0	5079	25.75	3.34
proxy3.rz.uni-karlsruhe.de	sibling	3128	3130	93459	0	10507	25.28	3.28
-		0	0	1411	0	2564	0.02	0.00

Tabla E.3 Refresh pattern efficiency

Refresh pattern	Options	No of req	Xfer (kB)	Req %	Xfer %	Xfer speed (kB/s)
^ftp:	1440 20% 10080	2399	4030139	0.22	25.52	16.45
^gopher:	1440 0% 1440	0	0	0.00	0.00	0.00
.	0 30% 8640	1091914	11759631	99.78	74.48	6.06

Tabla E.4 By MIME type

MIME type	No of requests	Xfer (kB)	Requests %	Xfer %	Xfer speed (kB/s)
text/plain	23384	4074718	2.14	25.81	17.50
application/octet-stream	8441	2424420	0.77	15.35	18.08
text/html	253526	2001903	23.17	12.68	5.12

image/jpeg	181058	1848041	16.55	11.70	15.37
image/gif	390117	940487	35.65	5.96	8.03
-	150781	862112	13.78	5.46	0.84
video/mpeg	1775	819444	0.16	5.19	28.77
video/x-msvideo	363	683222	0.03	4.33	42.09
application/x-zip-compressed	109	457537	0.01	2.90	35.32
audio/mpeg	198	286363	0.02	1.81	28.18
application/zip	245	175305	0.02	1.11	33.65
application/x-mpeg	69	139248	0.01	0.88	17.63
audio/x-pn-realaudio	195	135958	0.02	0.86	19.03
application/x-shockwave-flash	4283	110914	0.39	0.70	22.45
multipart/x-byteranges	96	106576	0.01	0.67	122.45
video/quicktime	45	95099	0.00	0.60	137.29
application/x-javascript	39321	92399	3.59	0.59	7.12
audio/x-mp3	5	71673	0.00	0.45	71.07
application/pdf	410	65721	0.04	0.42	16.06
application/x-debian-package	131	42472	0.01	0.27	36.73
application/x-tar	11	37445	0.00	0.24	78.68
multipart/byteranges	125	28434	0.01	0.18	113.33
video/x-ms-asf	142	27636	0.01	0.18	4.65
text/css	8940	22382	0.82	0.14	4.96
audio/x-ms-wma	31	20707	0.00	0.13	5.74
video/msvideo	13	19978	0.00	0.13	44.20

application/internet-property-stream	108	18426	0.01	0.12	18.37
video/x-ms-wmv	10	15792	0.00	0.10	27.88
application/x-msdos-program	16	15743	0.00	0.10	115.40
audio/x-mpeg	17	14780	0.00	0.09	118.98
application/x-bzip2	5	12193	0.00	0.08	8.90
text/xml	3750	11196	0.34	0.07	6.22
application/msword	37	10479	0.00	0.07	52.91
image/png	3094	9905	0.28	0.06	4.28
image/tiff	30	8602	0.00	0.05	2.31
image/pjpeg	316	7304	0.03	0.05	20.65
application/x-rar-compressed	12	6844	0.00	0.04	38.31
image/bmp	307	5836	0.03	0.04	50.55
aim/http	18107	5198	1.65	0.03	0.46
/	16	4463	0.00	0.03	128.83
application/macbinary	2	2880	0.00	0.02	39.87
application/x-rpm	1	2871	0.00	0.02	709.93
audio/basic	192	2852	0.02	0.02	31.81
audio/midi	136	2808	0.01	0.02	27.85
audio/x-realaudio	24	2556	0.00	0.02	14.03
application/jar	15	2294	0.00	0.01	433.17
application/x-gzip	8	2260	0.00	0.01	19.45
audio/x-wav	46	2102	0.00	0.01	15.79
application/postscript	7	1972	0.00	0.01	67.94

application/unix-tar	1	1922	0.00	0.01	88.69
application/realnetworksupgrade	9	1754	0.00	0.01	86.40
application/vnd.ms-powerpoint	4	1649	0.00	0.01	38.61
application/unknown	1	1613	0.00	0.01	13.34
application/x-director	11	1537	0.00	0.01	156.64
img/*	60	1299	0.01	0.01	16.70
x-compression/x-rar	2	1268	0.00	0.01	158.27
video/x-vcr	6	1216	0.00	0.01	33.61
application/wbload	6	1203	0.00	0.01	26.88
application/vnd.ms-excel	15	1132	0.00	0.01	106.43
message/rfc822	78	1091	0.01	0.01	14.79
application/x-compressed	149	1018	0.01	0.01	5.53
application/x-unknown-application-octet-stream	1	812	0.00	0.01	14.56
video/webride	2	751	0.00	0.00	4.89
application/x-shockwave-flash2-preview	7	734	0.00	0.00	155.42
audio/wav	11	668	0.00	0.00	22.60
image/jpg	48	640	0.00	0.00	24.99
application/x-stuffit	2	595	0.00	0.00	37.90
multipart/x-mixed-replace	3	594	0.00	0.00	3.22
application/x-msn-messenger	2048	578	0.19	0.00	1.49
application/x-pointplus	215	542	0.02	0.00	12.55
multipart/mixed	9	516	0.00	0.00	0.12
application/cab	5	480	0.00	0.00	294.69

application/x-java-archive	9	480	0.00	0.00	138.48
text/x-component	40	467	0.00	0.00	166.56
application/x-webshots	8	439	0.00	0.00	58.85
application/x-gzip-compressed	1	433	0.00	0.00	1.96
application/vnd.rn-realmedia	4	371	0.00	0.00	0.38
application/x-shswf	10	305	0.00	0.00	34.32
text/javascript	143	303	0.01	0.00	8.13
audio/mid	23	285	0.00	0.00	10.85
application/download	3	282	0.00	0.00	110.71
application/futuresplash	9	268	0.00	0.00	94.68
application/octetstream	23	216	0.00	0.00	1.75
application/x-ipix	3	202	0.00	0.00	171.13
www/unknown	32	186	0.00	0.00	6.37
application/x-chess-pgn	3	186	0.00	0.00	381.34
unknown/unknown	11	167	0.00	0.00	26.40
application/x-unknown-application-msword	1	143	0.00	0.00	16.47
text/axp	22	139	0.00	0.00	16.92
application/binary	11	114	0.00	0.00	119.50
image	106	105	0.01	0.00	5.04
unknown	137	100	0.01	0.00	2.30
application/java-vm	21	98	0.00	0.00	57.24
application/x-netgravity	202	97	0.02	0.00	3.68
application/vnd.rn-realplayer	84	90	0.01	0.00	0.69

image/jpg.png.gif	14	87	0.00	0.00	76.63
application/msexcel	4	71	0.00	0.00	15.34
application/x-octet-stream	1	64	0.00	0.00	14.52
image/*	4	57	0.00	0.00	2.10
text/http	42	48	0.00	0.00	7.46
application/x-java	8	48	0.00	0.00	34.38
application/x-zip	2	43	0.00	0.00	150.73
text/richtext	2	38	0.00	0.00	25.45
image/x-icon	19	35	0.00	0.00	6.40
java/*	3	28	0.00	0.00	70.08
audio/x-midi	1	28	0.00	0.00	33.89
images/gif	13	26	0.00	0.00	38.10
text/devil	87	25	0.01	0.00	0.89
app/x-hotbar-xip20	40	25	0.00	0.00	2.48
application/x-dvi	1	24	0.00	0.00	40.91
audio/x-pn-realaudio-plugin	56	21	0.01	0.00	1.19
image/bitmap	1	20	0.00	0.00	26.15
image/swf	2	20	0.00	0.00	2.41
application/metastream	2	18	0.00	0.00	40.05
application/x-jar	1	18	0.00	0.00	148.73
application/java	2	17	0.00	0.00	176.92
application/x-m3u	2	16	0.00	0.00	10.10
image/x-xbitmap	23	16	0.00	0.00	1.30

application/vnd.rn-rn_music_package	10	15	0.00	0.00	3.27
audio/x-scpls	23	14	0.00	0.00	2.72
image/unknown	1	14	0.00	0.00	28.06
application/pkix-crl	15	14	0.00	0.00	3.26
audio/x-mpegurl	33	14	0.00	0.00	1.19
application/java-archive	6	14	0.00	0.00	11.30
text/text	19	13	0.00	0.00	1.75
audio/x-aiff	1	12	0.00	0.00	347.26
application/vnd.ms-asf	2	12	0.00	0.00	4.54
image/"$expression"	2	11	0.00	0.00	6.61
application/kdedownload	1	10	0.00	0.00	102.99
application/x-java-vm	1	10	0.00	0.00	56.06
application/x-java-jnlp-file	22	8	0.00	0.00	1.95
application/x-httpd-cgi	17	7	0.00	0.00	0.56
allication/x-icon	1	7	0.00	0.00	20.08
img/gif	23	6	0.00	0.00	2.72
image/vnd.wap.wbmp	10	6	0.00	0.00	1.67
cnn/user-cookie	21	6	0.00	0.00	1.64
image/x-ms-bmp	1	5	0.00	0.00	96.68
img/png	2	5	0.00	0.00	1.67
javascript/text	4	5	0.00	0.00	0.69
image/x-bitmap	1	5	0.00	0.00	625.49
application/x-perl	12	4	0.00	0.00	0.40

image/x-png	1	4	0.00	0.00	1.09
application/smil	4	4	0.00	0.00	15.67
application/x-quicktime-response	4	3	0.00	0.00	2.14
magnus-internal/notfemail	6	3	0.00	0.00	0.89
video/x-la-asf	2	3	0.00	0.00	0.28
image/	6	3	0.00	0.00	1.14
application-x/javascript	3	3	0.00	0.00	4.61
application/x-icq	8	2	0.00	0.00	0.50
application/x-troff	2	2	0.00	0.00	7.65
text	5	2	0.00	0.00	1.02
:	6	2	0.00	0.00	5.83
application/x-httpd-php	2	2	0.00	0.00	29.94
application/x-wais-source	1	2	0.00	0.00	77.72
x-application/cnf	6	2	0.00	0.00	2.61
application/x-httpd-php3	1	2	0.00	0.00	26.19
application/x-java-applet	1	1	0.00	0.00	98.14
application/x-www-form-urlencoded	3	1	0.00	0.00	1.61
magnus-internal/cold-fusion	5	1	0.00	0.00	0.32
application/x-vermeer-rpc	3	1	0.00	0.00	1.71
text/x-vcard	2	1	0.00	0.00	2.08
video/x-ms-wvx	2	0	0.00	0.00	12.71
7am/clockfeed	5	0	0.00	0.00	0.57
application/x-mql	2	0	0.00	0.00	1.13

application/x-mplayer2	1	0	0.00	0.00	0.98
application/x-unknown	1	0	0.00	0.00	1.29
magnus-internal/jsp	1	0	0.00	0.00	2.06
text/phtml	1	0	0.00	0.00	1.36
application/vndms-pkiseccat	1	0	0.00	0.00	5.16
text/js	1	0	0.00	0.00	5.47

Tabla E.5 By cache result codes

Cache result code	No of req	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TCP_MEM_HIT	11809	1.08	75041	0.48	138.58
TCP_REFRESH_HIT	44485	4.07	272067	1.72	6.39
TCP_IMS_HIT	113829	10.40	29073	0.18	15.53
TCP_REFRESH_MISS	13775	1.26	109010	0.69	9.36
TCP_DENIED	51	0.00	56	0.00	160.77
TCP_CLIENT_REFRESH_MISS	69613	6.36	129333	0.82	4.59
TCP_HIT	134866	12.32	1632192	10.34	37.67
TCP_NEGATIVE_HIT	9173	0.84	9973	0.06	320.49
TCP_MISS	696699	63.67	13532976	85.71	6.58
TCP_SWAPFAIL_MISS	13	0.00	45	0.00	14.94

Tabla E.6 By peer status

Peer status	Req by fetch	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TIMEOUT_NONE	2564	0.23	1411	0.01	0.02
DIRECT	598652	54.71	10466791	66.29	6.86
TIMEOUT_DIRECT	192020	17.55	3347677	21.20	12.58
SIBLING_HIT	19745	1.80	209658	1.33	26.09
NONE	281332	25.71	1764232	11.17	5.52

Tabla E.7 Performance

User time	137.75 s
System time	6.89 s
Children time	0.00 s
Children system time	0.00 s
Together	**144.64** s
Performance	**7565 lines/s**

running on **Linux proxy4 2.2.19pre17 #1 Tue Mar 13 22:37:59 EST 2001 i686 unknown**

---

squeezer.pl v. 0.31 M.K., last modified: Mon Aug 16 13:00:21 1999

Tutorial on using squeezer generated data

ANEXO F

F.1 Reporte de Squeezer para el Servidor Proxy en la Universidad Nacional de Misiones

F.1.1 Correspondiente al período 18/08 – 28/08 de 2002

Tabla F.1 General information

Start date	Sun Aug 18 00:01:08 2002
End date	Wed Aug 28 10:33:26 2002
Total time	901938 s
HTTP requests per hour	2403
Transfer per hour	12152 kB

	No of req	Xfer (kB)	Xfer speed (kB/s)	Xfer %	Times to direct
Total traffic	602209	3044672	0.295	100.00	**1.11**
Direct fetches	358693	2559506	0.266	84.07	1.00
This server	240331	475520	0.775	15.62	2.91
Other servers	3185	8604	1.397	0.28	5.26
Cache hierarchy	243516	484124	0.781	15.90	2.94

Tabla F.2 Sibling efficiency

Server	Current relation and options	HTTP	ICP	Xfer (KB)	Xfer %	No of req	Xfer speed (KB/s)	Times to direct
170.210.193.20	sibling proxy-only	8080	3130	8604	0	1979	1.40	5.26
-		0	0	1041	0	1206	0.02	0.07

Tabla F.3 Refresh pattern efficiency

Refresh pattern	Options	No of req	Xfer (kB)	Req %	Xfer %	Xfer speed (kB/s)

Tabla F.4 By MIME type

MIME type	No of requests	Xfer (kB)	Requests %	Xfer %	Xfer speed (kB/s)
text/html	101714	772205	16.89	25.36	0.29
image/jpeg	88454	632545	14.69	20.78	0.47
image/gif	254023	557679	42.18	18.32	0.28
application/octet-stream	12544	198573	2.08	6.52	0.66
audio/mpeg	246	179963	0.04	5.91	2.22
application/zip	3831	140517	0.64	4.62	0.40
text/plain	12941	103852	2.15	3.41	0.36
application/x-shockwave-flash	4830	102856	0.80	3.38	0.63
application/pdf	889	96865	0.15	3.18	0.85
application/x-javascript	19595	43346	3.25	1.42	0.18

video/mpeg	210	42153	0.03	1.38	2.51
audio/x-pn-realaudio	82	37331	0.01	1.23	0.53
-	87837	28567	14.59	0.94	0.01
text/css	8408	17246	1.40	0.57	0.20
application/msword	103	14834	0.02	0.49	1.16
multipart/x-byteranges	135	11883	0.02	0.39	0.67
application/x-zip-compressed	28	9355	0.00	0.31	1.42
multipart/byteranges	87	6044	0.01	0.20	0.66
application/vnd.ms-powerpoint	11	5377	0.00	0.18	0.86
application/rtf	23	5345	0.00	0.18	2.12
interface/x-winamp3-skin	5	4319	0.00	0.14	13.20
video/quicktime	22	3848	0.00	0.13	0.57
video/x-ms-asf	49	3762	0.01	0.12	1.15
application/vnd.ms-excel	30	3484	0.00	0.11	4.32
audio/midi	201	2422	0.03	0.08	0.35
image/png	606	2247	0.10	0.07	0.30
application/x-stuffit	2	2107	0.00	0.07	6.48
image/pjpeg	19	1870	0.00	0.06	1.25
application/futuresplash	9	1322	0.00	0.04	1.09
application/x-msn-messenger	3445	1181	0.57	0.04	0.03
video/x-msvideo	6	1076	0.00	0.04	0.73
image/bmp	154	915	0.03	0.03	0.35
audio/mid	36	891	0.01	0.03	1.52

application/powerpoint	6	834	0.00	0.03	0.69
text/xml	278	772	0.05	0.03	0.08
audio/basic	28	717	0.00	0.02	2.27
audio/x-wav	31	695	0.01	0.02	1.39
application/x-director	5	686	0.00	0.02	0.44
application/mac-binhex40	1	602	0.00	0.02	18.44
text/rtf	11	541	0.00	0.02	0.17
application/x-pointplus	283	501	0.05	0.02	0.30
application/pkix-crl	32	457	0.01	0.02	0.58
application/x-msdos-program	177	356	0.03	0.01	0.26
text/x-component	41	332	0.01	0.01	1.95
message/rfc822	22	198	0.00	0.01	0.07
text/javascript	61	170	0.01	0.01	0.40
audio/x-ms-wma	7	154	0.00	0.01	7.08
application/bgl	1	139	0.00	0.00	0.19
application/x-compressed	31	135	0.01	0.00	0.73
image/x-bitmap	2	114	0.00	0.00	3.08
application/x-tar	2	109	0.00	0.00	0.05
audio/wav	1	106	0.00	0.00	0.93
application/xml	58	101	0.01	0.00	0.15
application/binary	7	98	0.00	0.00	0.80
application/x-java	8	86	0.00	0.00	0.61
text/devil	221	62	0.04	0.00	0.02

video/x-ms-wmv	2	56	0.00	0.00	0.55
application/x-ban	3	54	0.00	0.00	0.89
/	5	48	0.00	0.00	2.57
image/x-icon	27	47	0.00	0.00	0.13
application/x-futuresplash	1	46	0.00	0.00	1.85
application/x-java-vm	5	41	0.00	0.00	0.77
application/vnd.rn-realplayer	50	39	0.01	0.00	0.11
application/gzip	10	38	0.00	0.00	0.36
application/vndms-pkiseccat	116	37	0.02	0.00	0.04
application/java	4	34	0.00	0.00	2.54
application/x-excel	1	32	0.00	0.00	1.37
text/js	2	30	0.00	0.00	7.22
application/postscript	1	27	0.00	0.00	0.11
application/x-incredimail	3	23	0.00	0.00	0.12
audio/x-midi	5	22	0.00	0.00	0.18
application/java-archive	3	20	0.00	0.00	0.65
application/x-movieplayer	2	17	0.00	0.00	1.22
application/x-gzip	1	12	0.00	0.00	1.32
application/image	2	12	0.00	0.00	0.25
java/*	2	11	0.00	0.00	0.12
image/tiff	1	8	0.00	0.00	0.13
cnn/user-cookie	21	6	0.00	0.00	0.02
application/x-wais-source	10	6	0.00	0.00	0.15

image/x-xbitmap	5	5	0.00	0.00	0.13
application/x-vermeer-rpc	12	4	0.00	0.00	0.42
text/vnd.wap.wml	3	2	0.00	0.00	0.01
httpd/yahoo-send-as-is	4	1	0.00	0.00	0.17
application/applefile	2	1	0.00	0.00	0.05
unknown	2	1	0.00	0.00	1.09
audio/x-mpegurl	3	1	0.00	0.00	0.06
application/x-comet-log	2	0	0.00	0.00	0.00
app/fireclick.x-hint.1	1	0	0.00	0.00	0.46
application/x-httpd-cgi	1	0	0.00	0.00	0.05
application/x-icq	1	0	0.00	0.00	0.18
audio/x-pn-realaudio-plugin	1	0	0.00	0.00	0.29
httpd/unix-directory	1	0	0.00	0.00	0.17
audio/vnd.rn-realaudio	1	0	0.00	0.00	0.04
application/x-perl	1	0	0.00	0.00	0.01
image/vnd.wap.wbmp	2	0	0.00	0.00	0.02
text/text	1	0	0.00	0.00	0.12

Tabla F.5 By cache result codes

Cache result code	No of req	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TCP_MEM_HIT	11147	1.85	24948	0.82	38.28
TCP_REFRESH_HIT	66401	11.03	228634	7.51	0.17
TCP_IMS_HIT	141962	23.57	34624	1.14	3.40
TCP_REFRESH_MISS	9005	1.50	45528	1.50	0.25
TCP_DENIED	86	0.01	82	0.00	26.09
TCP_CLIENT_REFRESH_MISS	41050	6.82	80074	2.63	0.11
TCP_HIT	66748	11.08	405361	13.31	1.66
TCP_NEGATIVE_HIT	4737	0.79	5375	0.18	14.43
TCP_MISS	261055	43.35	2220032	72.92	0.28
TCP_SWAPFAIL_MISS	18	0.00	8	0.00	0.09

Tabla F.6 By peer status

Peer status	Req by fetch	Req %	Xfer (kB)	Xfer %	Xfer speed kB/s
TIMEOUT_NONE	1206	0.20	1041	0.03	0.02
DIRECT	309932	51.47	2155795	70.81	0.26
TIMEOUT_DIRECT	48761	8.10	403710	13.26	0.30
SIBLING_HIT	1979	0.33	8604	0.28	1.40
NONE	240331	39.91	475520	15.62	0.77

Tabla F.7 Performance

User time	73.95 s
System time	4.29 s
Children time	0.00 s
Children system time	0.00 s
Together	**78.24** s
Performance	**7696 lines/s**

running on **Linux campus 2.2.19 #93 Thu Jun 21 01:09:03 PDT 2001 i686 unknown**

---

squeezer.pl v. 0.31 M.K., last modified: Fri Aug 16 07:46:27 2002

Tutorial on using squeezer generated data